# Notifications

notification

The notification system supports sending messages via:

- Email
- SMS
- Concole (logger) - mainly for testing purpose
- … other custom channel can be provided by modules

## Implemented notification types (channels)

Notification type tells, which channel (⇒ which notification sender) will be used, when notification with given type will be send.

- `email` - default sender `emailNotificationSender` (leads to `DefaultEmailNotificationSender` implementation).
- `sms` - no sender is configured by default. Sender implementation has to registred by custom module and then configured.
- `console` - default sender `consoleNotificationSender` (leads to `DefaultConsoleNotificationSender` implementation).

Sender names above can be used in sender configuration in `impl` property, this ones are configured by default.

### Notification system functions

- One message can be sent in more ways (e.g. according to the configuration of a user's account) or sent by a specific channel defined by notification type.
- Messages are saved into the database - there is a notification agenda available for the application administrator
- Sending messages can be disabled by configuration (e.g. for testing purposes)
- If the recipient of the given notification isn't entered (e.g. a blank email), a record of not sending the notification is done in the log with level information

A notification (with message) is represented by entity `IdmNotification`, which keeps information about the notification itself (from, `recipients`, cc, bcc, `subject`, `message`, sent date). From the entity `IdmNotification` can be inherited by individual more specific types of notifications (i.e. `IdmEmail`). A notification for the identity is sent via service `NotificationService`. It is also possible to use a specific way of sending the notificaction, e.g. via `EmailService`, if the application requires it (e.g. from workflow).

Sending of notificanions itself is done by `Apache Camel`. The producer (`ProducerTemplate`) registers the notification (via `NotificationService`) - the route setting deals with propagating of the notification to concrete consumers (`EmailService` …), which send the notification and make a record in the db (`IdmNotificationRepository`) and the log. Consumers can be implemented via `ConsumerTemplate` or spring beans (preferred spring bean).

Apache Camel is used not only for the above mentioned reasons but it also supports sending of emails, SMSs and jms.

# Emailer

A concrete implementation of the notification sender via email (a wrapping of the camel emailer). Emailer (or rather EmailNotificationSender) is also injected into the workflow engine in order to be able to turn off sending of notifications for testing purposes. It is also possible to send emails for a specific identity (recommended) by entering the user name instead of the recipient's (or sender's) email address.

## Emailer settings

The emailer can be configured via application.properties or application settings agenda. A functional testing emailer settings.

## Emailer testing

If you want to test some project specific feature with sending email you can use AbstractNotificationTest, this class allow setup your own SMTP server (port and etc.), class allow add your own implementation of Observer, or is possible to use this implementation: NotificationObserver with solved concurrent thread.

# Sms

Currently there is a support for sending sms messages from IdM, but administrator must provide concrete implementation of sender for particular sms gateway. This can be done by extending AbstractSmsNotificationSender and registering this implementation in application context.

# Sending notification and processing

Sending messages is done by NotificationManager, which inherits from NotificationSender.java. In class NotificationSender we can find overloaded methods send, which are used for sending notifications.

The priority of evaluation is as follows:

1. The text is filled directly into method send. Even if there was a template, the directly filled-in text is used
2. sending the template directly via IdmMessage into method send, overwrites the template used in configuration

3. sending the template via topic. The template will be substituted from `IdmNotificationConfiguration`

# Notification and attachments

Notification sender can send attachments. The only sender, which supports sending attachment is email sender now. Sent attachments are persisted automatically and it's possible to send them from backend only. Attachments (`IdmAttachmentDto`) can be instanced (`inputData` will be used) or can be persisted before - data will be loaded by `AttachmentManager`, when notification (email) is send (see `DefaultEmailer`).

# Configuration

## Notification sender

Senders are configurable thanks to interface `Configurable` via the standard application configuration. More than one implementation of notification sender for one notification type can be installed into CzechIdM. We need to configure, which one is used, when notification with given type is send. Sender with the lowest order is used as default. Core senders using order **0**.

There can be more senders for the same notification type ⇒ the one that is going to be used can be chosen via configuration item `impl`. Any module can be registered in this way and also the behaviour of the existing senders, which are implemented via interface `NotificationSender`, can be changed.

## Examples

Everything is explained in the following examples:

**number 1.**

```
notificationManager.send(
    AccModuleDescriptor.TOPIC_NEW_PASSWORD,
    new IdmMessage.Builder()
        .setLevel(NotificationLevel.SUCCESS)
        .addParameter("systemName",
provisioningOperation.getSystem().getName())
        .addParameter("uid", provisioningOperation.getSystemEntityUid())
        .addParameter("password", password)
        .setSubject("I cannot be stopped")
        .setMessage("Hi, I'll overwrite everything!!")
        .build(),
```

```
    identity);
```

**number 2.**

```
notificationManager.send(
    AccModuleDescriptor.TOPIC_NEW_PASSWORD,
    new IdmMessage.Builder()
        .setLevel(NotificationLevel.SUCCESS)
        .addParameter("systemName",
provisioningOperation.getSystem().getName())
        .addParameter("uid", provisioningOperation.getSystemEntityUid())
        .addParameter("password", password)
        .setTemplate(template)
        .build(),
    identity);
```

**number 3.**

```
notificationManager.send(
    AccModuleDescriptor.TOPIC_NEW_PASSWORD,
    new IdmMessage.Builder()
        .setLevel(NotificationLevel.SUCCESS)
        .addParameter("systemName",
provisioningOperation.getSystem().getName())
        .addParameter("uid", provisioningOperation.getSystemEntityUid())
        .addParameter("password", password)
        .build(),
    identity);
```

**number 4.**

```
IdmAttachmentDto attachment = new IdmAttachmentDto();
attachment.setName("rest2.txt");
attachment.setInputData(IOUtils.toInputStream("test txt content 1234567899
ě+ščřžýáíéáýžřčšě+;ěščřžýáíééů", AttachableEntity.DEFAULT_CHARSET));
attachment.setEncoding(AttachableEntity.DEFAULT_ENCODING);
attachment.setMimetype("text/plain");

notificationManager.send(
    AccModuleDescriptor.TOPIC_NEW_PASSWORD,
    new IdmMessage.Builder()
        .setLevel(NotificationLevel.SUCCESS)
        .addParameter("systemName",
provisioningOperation.getSystem().getName())
        .addParameter("uid", provisioningOperation.getSystemEntityUid())
```

```
        .addParameter("password", password)
        .build(),
    null,
    Lists.newArrayList(identity),
    Lists.newArrayList(attachment)
);
```

If there is no message and template, the notification will not be sent. A notification log will be saved instead saying that message content doesn't exist.

### Disable sending notifications

Feature notification's configuration disabled attribute was added. While sending notification, notification configurations are searched and IdM searches for suitabled ones. Firstly it searches for exact topic and level, if it did not find any, it continue search just with topic and even then if no notification configuration is found, notification will be send anyway. To this algorithm disabled option was added. So now if notification configurations are found, just these, which are disabled are not send:

1. search exact topic, level → if found, send just to not disabled ones and not continue. if not found continue with 2.
2. search just with topic for general notification configuration → if not found, send anonymously, if found, check if it not diasbled.

### Add Recipients or redirect sending notifications

Into notification's configuration detail additional two items was added. It is 'Recipients' and 'Redirect'. Into Recipients you can add email adresses seperated with comma. Notifications with this topic will be send to normal recipient (e.g. password is changed for John identity → notification sent to John), and also it'll be sent to addresses specified in 'Recipients'. The second item 'Redirect' means, notification will only be sent to recipients specified in 'Recipients' and 'Recipients' field is mandatory, when 'Redirect' is selected.

## Future developement

- It will be possible to configure a testing receiver, where notifications will be sent to instead of the original one.
- Sending a notification will be tied to the configuration of the given entity, which will give the way of sending the notification (channel).

From:
https://wiki.czechidm.com/ - **IdStory Identity Manager**

Permanent link:
**https://wiki.czechidm.com/devel/documentation/notifications/dev/notification_manager**

Last update: **2020/09/16 09:10**