What's new in JHipster v2.0

Index

New directory layout

We have copied the directory layout from https://github.com/DaftMonk/generator-angular-fullstack , which is well-known for its high quality.

We now have, in *src/main/webapp* :

- *assets/* for all static assets such as images, fonts and CSS styles
- *bower_components/* for all installed Bower components
- *i18n/* for internationalization files
- *scripts/app* in which we have modules for each part of the application. When you add a new entity, it will have its AngularJS router, controller and HTML template in the same *scripts/app/entities/example* module. This will allow better modularity for bigger projects
- *scripts/components* for scripts which are common to several modules: for example navigation, authentication or internationalization

ui-router is the new router

We have switched from the default ng-router to https://github.com/angular-ui/ui-router as it is more flexible and is more powerful when handling partial views.

Improved i18n support

During the generation of a new project, JHipster only installs English and French languages.

However, JHipster supports more languages and that can be installed using this sub-generator.

Which languages are supported out of the box?

- Catalan
- Chinese (Traditional)
- Danish
- German
- Korean
- Polish
- Portuguese (Brazilian)
- Russian
- Spanish
- Swedish
- Turkish

How to install new languages?

In order to install new languages, just type:

yo jhipster:languages

How to create a new language that is not supported?

All languages are saved in the folder */src/main/webapp/i18n*

Here are the steps to install a new language called *new_lang*:

- Duplicate the *<code>/src/main/webapp/i18/en* folder to */src/main/webapp/i18/new_lang*
- Translate all files under the folder */src/main/webapp/i18/new_lang*
- Update the *LANGUAGES* constant defined in the folder *src/main/webapp/components/language/language.service.js* to add the new language *new_lang*

```
.constant('LANGUAGES', [
  'en', 'fr', 'new_lang'
  //JHipster will add new languages here
]
```

The new language *new_lang* is now available in the language menu

“yo angular” support

All commands from https://github.com/yeoman/generator-angular now work in JHipster, which gives you a lot of small sub-generators for specific needs.

For example, you can call “yo angular:decorator mydecorator” or “yo angular:directive mydirective”.

As this is new and adds a lot of new possibilities, please provide feedback if you find some interesting usages!

Liquibase changes

We now support https://github.com/liquibase/liquibase-hibernate. This allows a new, more efficient workflow when modifying an existing entity:

1. Modify the entity (for example, add a new field)

2. Run “mvn compile liquibase:diff”
    a. Or run “./gradlew liquibaseDiffChangelog” when using Gradle
3. This will generate a new changelog in your ‘src/main/resources/config/liquibase/changelog’ folder
4. You need to add this new changelog to your ‘src/main/resources/config/liquibase/master.xml’ file
5. Next time you start up your application, this changelog will be applied to the database, and thus your entity will work correctly

Liquibase Hibernate is a Maven plugin that is configured in your pom.xml, and is independent from your Spring application.yml file, so if you have changed the default settings (for example, changed the database password), then both files need to be modified.

This plugin only works when you have a running database against which it can make a diff, so it only works if you are using MySQL or Postgresql in development (and not H2, as it is running in-memory with your application, and it is not available outside your application).

When using gradle change the database configuration in liquibase.gradle if required.

Better template caching

Thanks to grunt-angular-template, we have much better caching of our static resources (see the PR: https://github.com/jhipster/generator-jhipster/pull/845), which will give a much improved performance in production.
Of course, it only works with the Grunt build.

The User domain object now has an “id”

The *User* domain object used the login as its PK, as this is the default schema from Spring Security. This was a bad idea, as:

- An ID should be a technical item, not a business one, as business logic can change (in this example, you might want to change your login)
- All other entities (including those from the entity sub-generator) use an “id” as a PK

So the *User* domain object now uses an “id” as a PK, and the “login” is just a standard field (with an index, of course!).

Spring Websocket support

Our Websocket option used to work with Atmosphere, but we had lots of integration issues between Spring Boot and Atmosphere.
So we migrated from Atmosphere to Spring Websocket, and as a result:

- This resolved all integration issues
- Our resulting code is much smaller

As we believe Atmosphere is a very robust and mature solution, our end goal is to support both Spring Websocket and Atmosphere, but for the first v2.0.0 release we will only support Spring Websocket.

Automatic generation of entities from a JSON file

When an entity is generated, its configuration options are now stored in a ".jhipster.entity.json" file.
This allows new automatic usage of the "entity" sub-generator:

- You can ask it to re-generate an existing entity: it can be useful if you modified your entity, or if you updated the generator
- Several people have already started working on scripting the sub-generator: you could auto-generate entities from an external tool, such as an UML or a database designer

New "entity detail" screen

In addition to the standard "entities" screen, the entity sub-generator now generates an "entity detail" screen, where you can view a single entity.

Metrics Spark reporting

We now provide a Spark reporter for Metrics, in addition to the already existing JMX and Graphite reporters.
With the Spark reporter, you can analyse your Metrics data on-the-fly with Spark Streaming. The Metrics Spark reporter is available at https://github.com/ippontech/metrics-spark-reporter

Statistics

We now gather anonymous statistics on the generator with https://github.com/yeoman/insight

Of course, you will get prompted if you want to send those statistics or not. They are fully anonymous (we send data like the build tool or the database you use, but not your package name or application name).

This data is very useful, it will be used by the JHipster team to know on which options or tools they need to focus on.