Integration tests

Manual tests

Yes, we have "manual tests" in something called "integration tests". These actually are integration tests (we're testing a bundled system, not independent components), they're simply not automated for various reasons: complexity of implementation of the automation vs resources availability vs maintainability. Hopefully some of these tests will be more easily automatable once we revamp the GUI components.

These tests should be executed at least once per release. The procedure to execute them is described at
Manual tests using the integration-tests webapp [Internal, Outdated].

To add such manual tests:

- clone https://git.magnolia-cms.com/scm/platform/ce.pub.git
- depending on what needs to be tested, you can do a combination of:
    - add templates in `magnolia-integration-tests-fixture-module/src/main/resources/mgnl-files/templates/test/manual`
    - declare those templates in `magnolia-integration-tests-fixture-module/src/main/resources/info/magnolia/test/config.properties` or via `info.magnolia.test.SetupStuffForTests`.
    - in the template itself, describe what actions need to be performed for the test to be considered successful
    - add a page with this template to the website, under `/manual-tests`, via `magnolia-integration-tests-fixture-module/src/main/resources/info/magnolia/test/website.properties`.

⚠️ please take this into consideration if you find yourself creating many templates that have no purpose other than holding the test description: in the future, the current `website.properties` file might be specialized (i.e use a custom, simpler, format, instead of the properties import format; additionally, the test description might figure in there instead of the template, especially for tests where templating isn't relevant); if you're in such case, feel free to do so yourself or give us a shout!

We have also provided "integration tests" for REST.

- Check out - https://git.magnolia-cms.com/scm/modules/rest.git
- We use Rest Assured to easy to test our API. See more syntax in its documentation https://github.com/rest-assured/rest-assured
- Add new integration tests under the path `/magnolia-rest-integration-tests/tests/src/test/java`
- Launch the container by using "manual-test" profile: `mvn clean verify -Pjetty9-standalone,manual-tests`
- Want to put some bootstrap dummy data? Put them under the folder `/magnolia-rest-integration-tests/magnolia-rest-test-webapp/src/main/webapp/WEB-INF/bootstrap/common`
- In order to add more endpoints or override endpoint configuration. Find the configuration in the folder `/magnolia-rest-integration-tests/magnolia-rest-test-webapp/src/main/webapp/modules/rest-content-delivery/restEndpoints`

Automated tests

We also have a suite of automated tests. They rely on the same process, but are executed differently:

- since `magnolia-integration-tests` lies inside the CE bundle project, they are executed every time to bundle is built. This means they're executed on a regular basis by Jenkins (https://jenkins.magnolia-cms.com/job/magnolia-bundle_trunk/) and every time we do a release.

To add automated tests, the procedure is similar to the above, but:

- if you add website pages, you'll want to add them under `/testpages`
- you need to write code to execute the test; the current tests reside in `magnolia-integration-tests/tests/src/test`. `info.magnolia.integrationtests.AbstractMagnoliaIntegrationTest` is meant to be a base class such tests could extend.

There are also groovy scripts in `magnolia-integration-tests/tests/src/main/`:

- Testing of activation [Internal] that activates content and checks for results
- Crawling test [Internal] that requests all the pages and checks for rendering exceptions
- Log crawling test that reads logs and checks for rendering exceptions.

REST integration tests will be executed everytime the REST module is built - https://jenkins.magnolia-cms.com/job/m_rest/configure

Concurrent builds for bundles on Jenkins

Integration tests run concurrently on Jenkins for CE and EE bundles, and for our various branches (5.0.x up to master). Therefore each branch/bundle must use its own ports for starting up Jetty and e.g. sending shutdown signals through the Cargo wrapper.

Here are the ports in use for our current builds:

	CE bundle			EE bundle		
	containerHttpPort	cargoRmiPort	cargoTomcatAjpPort	containerHttpPort	cargoRmiPort	cargoTomcatAjpPort
bundle-4.5.x	8067	8066		9067	9066	
bundle-5.0.x	8077	8076		-	-	
bundle-5.1.x	8099	8098		9099	9098	
bundle-5.2.x	8199	8198		9199	9198	
bundle-5.3.x	8299	8298		9299	9298	
bundle-5.4.x	8399	8398	8397	9399	9398	9397
bundle-5.5.x	8499	8498	8497	9499	9498	9497
bundle-master (5.6)	8599	8598	8597	9599	9598	9597
teams/platform	7529	7528		-	-	
teams/cms	8267	8266		9267	9266	
teams/integration	8468	8467		9468	9467	
bundle-debug	6667	6666	6665	7771	7772	7773
bundle-5.4.x-cp13n	-	-		9390	9389	9388

Configuration for these ports is located in:

- `magnolia-integration-tests/tests/pom.xml` , properties `<containerHttpPort>` and `<cargoRmiPort>`
  - automated tests, manual tests for IntelliJ IDEA
- `info.magnolia.testframework.AbstractMagnoliaIntegrationTest => DEFAULT_DOMAIN`
  - manual tests for eclipse

⚠️ When branching a bundle's master to the next major version, one should update these ports according to the pattern in the table above.

Since 2.0, REST integration tests also run on the cargo/test-webapp setup, therefore they also have to use unique ports:

	CE bundle		
	containerHttpPort	cargoRmiPort	cargoTomcatAjpPort
release/2.0	8396	8395	8394
release/2.1	8496	8495	8494

These ports are configured in: `/rest/magnolia-rest-integration-tests/tests/pom.xml`

Debugging

To debug these tests, one way to go is to start the container the same way as for the manual tests, with `mvn clean verify -Pmanual-tests`, and run the tests from the IDE in debug mode.
Somehow, memory issues seem to arise, at least with EE, in the manual-tests mode. This seemed to have helped, not sure which statement exactly:

```
export JAVA_OPTS="-Xms64M -Xmx1024M -Djava.awt.headless=true"
export JAVA_OPTIONS="-Xms64M -Xmx1024M -Djava.awt.headless=true"
export JETTY_OPTIONS="-Xms64M -Xmx1024M -Djava.awt.headless=true"
export JETTY_OPTS="-Xms64M -Xmx1024M -Djava.awt.headless=true"
```

Generalities

All these tests are executed within the context of a single Jetty instance, which is started by the Cargo plugin of Maven.

The current setup deploys two Magnolia instances: `magnolia-test-webapp` and `magnolia-test-webapp-public`, under the `/magnoliaTest` and `/magnoliaTestPublic` contexts, respectively.
Both are an extension of `magnolia-empty-webapp`, additionally installing the `magnolia-templating-samples` and `magnolia-integration-tests-fixture-module`, which contains the pages, templates etc created above, for use while executing the tests.

If more instances or different setups would be needed, we would probably have to extend this rather simple architecture quite a bit, at which point it will probably become unpractical to run these within the bundle; we'll have to figure out a procedure to ensure those tests are still systematically executed WITH the release being made.