LORE Documentation

Release 0.16.0

MIT Office of Digital Learning

March 23, 2016

Contents

CHAPTER 1

Getting Started

You can either run this locally with a default sqlite database after installing the requirements.txt file, or if you have Docker and prefer a cleaner environment, install docker-compose with `pip install docker-compose` and run `docker-compose up`. This will set up a near production-ready containerized development environment that runs migrations, with the django development server running on port 8070.

To run one-off commands, like shell, you can run `docker-compose run web python manage.py shell` or to create root user, etc.

Currently in the development environment we compile JSX on every request, which at the time of this writing is about 5 seconds. If you want to disable this (because you are working just on python for example), you can add the line `LORE_COMPRESS_ENABLED: True` to *docker-compose.yml* under the web -> environment section of the file. The first request will then take 5 seconds, but subsequent ones will be subsecond.

CHAPTER 2

Adding an application

To add an application to this, add it to the requirements file, add its needed settings, include its URLs, and provide any needed template overrides.

Adding JavaScript or CSS Libraries

We have bower installed and configured in the docker image. This is very handy for documenting and adding dependencies like backbone or bootstrap. To add a new dependency, just run `docker-compose run web bower install jquery --save` for example. This will download jquery to the `lore/static/bower/jquery` folder and add it to the bower.json file. The assets downloaded should be stripped down to as little as needed before checking in, but the files should be checked into the repository.

CHAPTER 4

Including JavaScript and CSS

We are using django-compressor for static asset compression, along with django-compressor-requirejs for creating requirejs packages. What this means to you is that you need to do static asset additions in your templates with something like:

```
{% load compress %}

{% compress css %}
<link rel="stylesheet"
  href="{% static "bower/bootstrap/dist/css/bootstrap.css" %}"
  type="text/css"
  charset="utf-8"
>
<style type="text/css">p { border:5px solid green;}</style>
{% endcompress %}

{% compress js %}
<script type="text/requirejs"
  src="{% static "bower/requirejs/require.js" %}">
</script>
<script type="text/javascript">
 require.config({
  baseUrl: '{% static "bower"%}',
  paths: {
      jquery: 'jquery/dist/jquery',
      bootstrap: 'bootstrap/dist/js/boostrap'
  }
  requirejs(["jquery"], function($) {
    .....
  }
</script>

{% endcompress %}
```

CHAPTER 5

Testing

The project is setup with tox and py.test. It will run pylint, pep8, and py.test tests with coverage. It will also generate an HTML coverage report. To run them all inside the docker image, run `docker-compose run web tox`, or if you are running locally, after installing the requirements file, just run `tox`.

The project also contains JavaScript tests which can be run using [Karma](karma-runner.github.io). `tox` will run the JavaScript tests after the Python tests. You can run only the JavaScript tests using `docker-compose run web tox -e js`, or do continuous JavaScript testing with `docker-compose -f docker-karma.yml up` and connecting to port 9876 on your docker host.

In addition to local testing, all commits and pull requests are tested on travis-ci.org.

5.1 Continuous Testing

If you want test to run on file changes, the `test_requirements.txt` adds pytest-watcher, which can be started with: `docker-compose run web ptw --poll` For additional options like having it say "passed" out loud, or sending desktop notifications for failures see the README. Keep in mind, there can be a bit of a lag between saves and the test running.

CHAPTER 6

Building Local Documentation

To build the sphinx documentation project, run: `docker-compose run web tox -e docs`

This will build it on your local machine and you should be able to point your browser at `</path/to/lore/repo>/docs/_build/index.html`.

CHAPTER 7

RESTful API Documentation

LORE has a RESTful API that is documented on Apiary http://docs.lore.apiary.io .

CHAPTER 8

Welcome to LORE's documentation!

Contents:

8.1 LORE API Docs

For convenient reference in development, here are the LORE API docs.

8.1.1 LearningResources

Helper functions for using the models, so external apps don't tie functionality to internal implementation.

exception `learningresources.api.`**`LearningResourceException`**

Bases: `exceptions.Exception`

Base class for our custom exceptions.

class `learningresources.api.`**`MissingTitle`**

Bases: `object`

Class to describe the missing title for importer and for the description path

`for_desc_path_field` **= u'...'**

`for_title_field` **= u'Missing Title'**

exception `learningresources.api.`**`NotFound`**

Bases: *`learningresources.api.LearningResourceException`*

Raised by the API when the item requested does not exist.

exception `learningresources.api.`**`PermissionDenied`**

Bases: *`learningresources.api.LearningResourceException`*

Raised by the API when the requested item exists, but the user is not allowed to access it.

`learningresources.api.`**`create_course`**(*org*, *repo_id*, *course_number*, *run*, *user_id*)

Add a course to the database.

Parameters

- **`org`** (*unicode*) – Organization
- **`repo_id`** (*int*) – Repository id
- **`course_number`** (*unicode*) – Course number

- **run** (*unicode*) – Run
- **user_id** (*int*) – Primary key of user creating the course

Raises `ValueError` – Duplicate course

Returns **course** – The created course

Return type learningresources.models.Course

`learningresources.api.`**`create_repo`**`(`*name*, *description*, *user_id*`)`

Create a new repository.

Parameters

- **name** (*unicode*) – Repository name
- **description** (*unicode*) – Repository description
- **user_id** (*int*) – User ID of repository creator

Returns **repo** – Newly-created repository

Return type learningresources.Repository

`learningresources.api.`**`create_resource`**`(`*course*, *parent*, *resource_type*, *title*, *content_xml*, *mpath*, *url_name*, *dpath*`)`

Create a learning resource.

Parameters

- **course** (*learningresources.models.Course*) – Course
- **parent** (*learningresources.models.LearningResource*) – Parent LearningResource
- **resource_type** (*unicode*) – Name of LearningResourceType
- **title** (*unicode*) – Title of resource
- **content_xml** (*unicode*) – XML
- **mpath** (*unicode*) – Materialized path
- **url_name** (*unicode*) – Resource identifier
- **dpath** (*unicode*) – Description path

Returns **resource** – New LearningResource

Return type learningresources.models.LearningResource

`learningresources.api.`**`create_static_asset`**`(`*course_id*, *handle*`)`

Create a static asset. :param course_id: learningresources.models.Course pk :type course_id: int :param handle: file handle :type handle: django.core.files.File

Returns learningresources.models.StaticAsset

`learningresources.api.`**`get_repo`**`(`*repo_slug*, *user_id*`)`

Get repository for a user if s/he can access it. Returns a repository object if it exists or * raises a 404 if the object does not exist * raises a 403 if the object exists but the user doesn't have permission

Parameters

- **repo_slug** (*unicode*) – Repository slug
- **user_id** (*int*) – Primary key of user

Returns **repo** – Repository

Return type learningresource.Repository `learningresources.api.`**`get_repos`**`(`*user_id*`)`

Get all repositories a user may access.

Parameters **`user_id`** (*int*) – Primary key of user

Returns **repos** – Repositories

Return type query set of learningresource.Repository

`learningresources.api.`**`get_resource`**`(`*resource_id*, *user_id*`)`

Get single resource.

Parameters

- **`resource_id`** (*int*) – Primary key of the LearningResource
- **`user_id`** (*int*) – Primary key of the user requesting the resource

Returns **resource** – Resource May be None if the resource does not exist or the user does not have permissions.

Return type learningresources.LearningResource

`learningresources.api.`**`get_resources`**`(`*repo_id*`)`

Get resources from a repository ordered by title.

Parameters **`repo_id`** (*int*) – Primary key of the repository

Returns **list** – List of resources

Return type list of learningresources.LearningResource

`learningresources.api.`**`get_video_sub`**`(`*xml*`)`

Get subtitle IDs from <video> XML.

Parameters **`xml`** (*lxml.etree*) – xml for a LearningResource

Returns subtitle string

Return type sub string

`learningresources.api.`**`import_static_assets`**`(`*course*, *path*`)`

Upload all assets and create model records of them for a given course and path.

Parameters

- **`course`** (*learningresources.models.Course*) – Course to add assets to.
- **`path`** (*unicode*) – course specific path to extracted OLX tree.

Returns None

`learningresources.api.`**`join_description_paths`**`(`**args*`)`

Helper function to format the description path. :param args: description path :type args: unicode

Returns Formatted dpath

Return type unicode

`learningresources.api.`**`type_id_by_name`**`(`*name*`)`

Get or create a LearningResourceType by name.

This would do fewer queries if it did all the lookups up front, but this is simpler to read and understand and still prevents most lookups. Also, it can't prevent inserts, so it's never guaranteed to be just a single query.

Parameters **`name`** (*unicode*) – LearningResourceType.name

Returns **type_id** – Primary key of learningresources.LearningResourceType

Return type int

`learningresources.api.`**`update_description_path`**`(`*resource*, *force_parent_update=False*`)`

Updates the specified learning resource description path based on the current title and the parent's description path :param resource: LearningResource :type resource: learningresources.models.LearningResource :param force_parent_update: force parent update :type force_parent_update: boolean

Returns None

`learningresources.api.`**`update_xanalytics`**`(`*data*`)`

Update xanalytics fields for a LearningResource. :param data: dict from JSON file from xanalytics :type data: dict

Returns **count** – number of records updated

Return type int

8.1.2 Importer

Import OLX data into LORE.

`importer.api.`**`import_children`**`(`*course*, *element*, *parent*, *parent_dpath*`)`

Create LearningResource instances for each element of an XML tree.

Parameters

- **`course`** (*learningresources.models.Course*) – Course
- **`element`** (*lxml.etree*) – XML element within xbundle
- **`parent`** (*learningresources.models.LearningResource*) – Parent LearningResource
- **`parent_dpath`** (*unicode*) – parent description path

Returns None

`importer.api.`**`import_course`**`(`*bundle*, *repo_id*, *user_id*, *static_dir*`)`

Import a course from an XBundle object.

Parameters

- **`bundle`** (*xbundle.XBundle*) – Course as xbundle XML
- **`repo_id`** (*int*) – Primary key of repository course belongs to
- **`user_id`** (*int*) – Primary key of Django user doing the import
- **`static_dir`** (*unicode*) – location of static files

Returns learningresources.models.Course

`importer.api.`**`import_course_from_file`**`(`*filename*, *repo_id*, *user_id*`)`

Import OLX archive from .zip or tar.gz.

Imports from a file and then deletes that file. A valid OLX archive has a single occurrence of the file course.xml in its root directory, or no course.xml in its root and a single occurrence of course.xml in one or more of the root directory's children.

Parameters

- **`filename`** (*unicode*) – Path to archive file (zip or .tar.gz)
- **`repo_id`** (*int*) – Primary key of repository course belongs to
- **`user_id`** (*int*) – Primary key of user importing the course

Returns None

Raises `ValueError` – Unable to extract or read archive contents.

`importer.api.`**`import_course_from_path`**(*path*, *repo_id*, *user_id*)

Import course from an OLX directory.

Parameters

- **`path`** (*unicode*) – Path to extracted OLX tree
- **`repo_id`** (*int*) – Primary key of repository course belongs to
- **`user_id`** (*int*) – Primary key of Django user doing the import

Returns course (learningresources.Course)

`importer.api.`**`is_leaf_tag`**(*tag*)

Should we look for resources within elements with this tag?

Parameters **`tag`** (*unicode*) – Element tag

Returns Whether tag is leaf tag

Return type bool

8.1.3 Taxonomy

APIs for lore taxonomy application

`taxonomy.api.`**`get_term`**(*repo_slug*, *user_id*, *vocab_slug*, *term_slug*)

Get Term with existing slug, validating ownership for repo_slug and vocab_slug.

Parameters **`term_id`** (*int*) – Term slug

Returns **Term** – The Term with the id

Return type Term

`taxonomy.api.`**`get_vocabulary`**(*repo_slug*, *user_id*, *vocab_slug*)

Lookup vocabulary given its slug, using repo_slug to validate ownership.

Parameters

- **`repo_id`** (*int*) – Repository id
- **`user_id`** (*int*) – User id
- **`vocab_slug`** (*unicode*) – Vocabulary slug

Returns **Vocabulary** – The vocabulary from the database

Return type Vocabulary

8.1.4 Membership

Functions for handling roles.

`roles.api.`**`assign_user_to_repo_group`**(*user*, *repo*, *group_type*)

Assign an user to a repo specific group type.

Parameters

- **`user`** (*django.contrib.auth.models.User*) – user

- **repo** (*learningresources.models.Repository*) – repository used to extract the right group to use
- **group_type** (*roles.permissions.GroupTypes*) – group string to be used to construct the group name

Returns None

`roles.api.`**`is_last_admin_in_repo`**`(`*user*, *repo*`)`

Check if user is the last administrator in the repository. It does not check if the user is an actual administrator and in that case it will simply return False

Parameters

- **user** (*django.contrib.auth.models.User*) – user
- **repo** (*learningresources.models.Repository*) – repository used to extract the right group to use

Returns bool

`roles.api.`**`list_users_in_repo`**`(`*repo*, *base_group_type=None*`)`

List all the users in the repository groups. If the group type is specified, the list is limited to that group.

Parameters

- **repo** (*learningresources.models.Repository*) – repository used to extract the right group to use
- **base_group_type** (*unicode*) – group type from roles.permissions.BaseGroupTypes

Returns **list** – list of users in one or all the repository groups

Return type list of roles.user_models.UserGroup

`roles.api.`**`remove_user_from_repo_group`**`(`*user*, *repo*, *group_type*`)`

Remove an user from a repository specific group type.

Parameters

- **user** (*django.contrib.auth.models.User*) – user
- **repo** (*learningresources.models.Repository*) – repository used to extract the right group to use
- **group_type** (*roles.permissions.GroupTypes*) – group string to be used to construct the group name

Returns None

`roles.api.`**`roles_clear_repo_permissions`**`(`*repo*`)`

Remove all the permissions a group has on a repository. :param repo: repository :type repo: learningresources.models.Repository

Returns None

`roles.api.`**`roles_init_new_repo`**`(`*repo*`)`

Create new groups for the repository.

It assumes that there are only 3 types of users:

- administrator
- curator
- author

Parameters **`repo`** (*learningresources.models.Repository*) – repository used to create groups and assign permissions to them

Returns None

`roles.api.`**`roles_update_repo`**(*repo*, *old_slug*)

Update the groups names for the repository.

Parameters

- **`repo`** (*learningresources.models.Repository*) – repository used to update groups and assign permissions to them
- **`old_slug`** (*unicode*) – old slug string used to retrieve the groups that need to be renamed

Returns None

8.2 Release Notes

8.2.1 Version 0.16.0

- Sorted vocabularies in api as well as taxonomy panel
- Replaced the status app with django-server-status
- Added warning message when user try to close learning resource panel without saving
- Updated Django requirement to latest security release
- ISSUE #876 Fixed doc build failure

8.2.2 Version 0.15.1

- Version locked `doc_requirements` packages.

8.2.3 Version 0.15.0

- Added sorting by relevance `(_score)`.
- Fixed requests installation.
- Fixed new pylint 1.5.0 violations.

8.2.4 Version 0.14.1

- Made `[requires.io]` dependency update.
- Added manual test plan.
- Made changes to accommodate Docker 1.9.0.

8.2.5 Version 0.14.0

- Moved spinner into separate React class.
- Fixed double resource import bug.
- Fixed preview link to use parent resource link if child has none.
- Fixed Elasticsearch in docker-compose.
- Implemented tracking import tasks.
- Added parsing for `display_name` so we can use it if `url_name` isn't present.
- Removed `django-compressor-requirejs` requirement.
- Moved taxonomy React components to separate files.
- Moved manage taxonomy tests into separate files.
- Moved React components to separate files in new directories.
- Moved tests into separate files.
- Added tests to verify index on course delete.
- Fixed intermittent test failures.
- Added compressor cache.
- Bound demo branch to lore-demo app deployment.
- Sorted terms in taxonomy panel.
- Renamed `listing.js` to JSX.
- Added tasks API.

8.2.6 Version 0.13.0

- Refactored listing code for testing.
- Implemented lazy loading for resource tab.
- Added custom slugify function to allow any name for Repo, Vocab, Term.
- Fixed index mapping of terms and vocabularies.
- Set Elasticsearch log level higher during testing.
- Set Spinner `zIndex` to 0 to not have it float above everything else.
- Added loader on save.
- Removed unused functions.
- Fixed link click behavior for term edit and delete.

8.2.7 Version 0.12.0

- Added task to index resources via Celery.
- Wrapped course imports in transaction.
- Limited vocabularies to repository being searched.

- Updated Django to 1.8.5 due to a bugfix release.
- Removed Haystack.
- Moved Pagination inside Listing React component.
- Fixed reindexing for edited vocabularies.
- Updated status tests to use settings context.
- Added limit to memory for Docker.
- Added loader to listing, resource panel and taxonomy panel.
- Chunked bulk indexing to lower memory footprint.
- Fixed resource ids.
- Added debug_toolbar.
- Fixed mouse pointer on edit and delete links.
- Fixed memory usage in migration.
- Fixed collapse behavior.
- Fixed NewRelic not reporting.
- Added bulk insert of static assets during import.
- Changed `get_or_create` to `create`.
- Replaced Haystack queryset and facet counts with elasticsearch-dsl.
- Fixed `id` attribute for vocab and term select.
- Added filtering on vocabulary and learning resource type for learning resources.
- Configured NewRelic Python agent.
- Implemented verification of resource types on models.
- Static assets are served again using local storage.
- Fixed missing test module.
- Replaced default missing title and updated description path.
- Pinned PyTest to a version < 2.8.
- Wrapped description text to 2 lines and added expand collapse description functionality.
- Added vocabulary editing in the taxonomy panel.
- Added documentation for LORE release workflow.

8.2.8 Version 0.11.0

- Reorganized Learning Resource panel to use three tabs.
- Added datatable proof-of-concept.
- Added REST api view to get and delete a course.
- Added calls to `get_conn()` where `conn` is used implicitly.
- Added more detail to confirmation message for delete vocabulary.
- Moved `Save` button to right of the term for edit term inside taxonomy panel.

- Switched Django local storage to overwrite.
- Implemented `page_size` parameter to allow users to set page size.
- Fixed spacing between `edit` and `delete` buttons.
- Removed /node directory, and removed symlinks from node_modules.
- Added elasticsearch-dsl and added it alongside Haystack for now.
- Added `Save and Close` button to learning resource panel.
- Added tests for listing page.
- Removed lib/ from `.gitignore`.
- Switched to minimized javascript for libraries.
- Added REST API view to list courses in repository.
- Removed react-addons bower package, addons actually live in react package.
- Fixed pagination links.
- Increased requirejs timeout.

8.2.9 Version 0.10.1

- Fixed exact repository search bug.
- Fixed clear export bug.

8.2.10 Version 0.10.0

- Added listing refresh after taxonomy changes.
- Added React component for not tagged count.
- Added link in README.rst to RESTful API doc on Apiary.
- Point to specific version of xbundle.
- Point to v0.3.1 of xbundle on Github.
- Cleaned up form-based search code.
- Changed behavior to use AJAX calls for listing page updates.
- Fixed bug with sorting by title being case sensitive.
- Installed history.js.
- Added capability to facet by missing Vocabulary terms in REST API search.
- Added inline editing feature for terms in taxonomy panel.
- Added delete vocabulary in taxonomy panel.
- Added sorting by title.
- Added Roles module to Sphinx documentation.
- Updated export to preserve static asset path.
- Fixed serving of images in javascript tests.
- Updated apiary docs for recent changes to API.

- Added REST endpoint for search.
- Created React component for pagination.
- Formatted average grade as fixed width number.
- Changed member list refresh to happen after AJAX success.
- Refactored facet view as React component.
- Added URI.js.
- Fixed counter in learning resource exports panels header.
- Fixed ordering of javascript variables due to stricter JSHint rules.
- Disable SSL validation for a test which uses urltools.
- Revert #540, add migration to revert related data migration.
- Added travis-ci build notifications for Hipchat and Slack.
- Don’t compress dynamic JavaScript.
- Fixed migration to bulk create rows in through table.
- Refactored listing resources to use React.
- Added bootstrap as requirement for manage taxonomies.
- Optimized Dockerfile to reduce build times.
- Added support for free tagging for terms.
- [requires.io] dependency update.

8.2.11 Version 0.9.0

- Stripped caching out of vocabularies during indexing.
- Changed password hashing during tests.
- Updated third party requirements.
- Made better navigation of paging in search results.
- Made creator of a repo an admin during repo creation.
- Fixed static asset download for local servers.
- Added lazy loading of static asset information.
- Added icon for logout previously reverted.

8.2.12 Version 0.8.0

- Changed how vocabulary terms are applied to Learning Resources to use two dropdowns instead of a growing list of fields.
- Added deployment for release candidates.
- Added deploy button and app.json.
- Fixed caching bug.
- Fixed panel shade issue.

- Added base sorting field in case used sorting is working on same values.
- Removed response from PATCH on learning resource to aid in performance.
- Added configuration option and heroku command to pre-compress assets.
- Added Google Analytics tracking support Closes.
- Reduce workers per dyno to avoid memory issues.
- Added statsd and a few timers.
- Updated indexing caching from dict to Django's cache.
- .tile-meta no longer defined twice.
- Split builds and removed python 3.3 testing.
- reverted tile-meta and meta-item for previous appearance.
- Added import for (sample) xanalytics API data.
- Added closing panels with ESC key.
- Fixed export button to show up even without search results.
- Updated CSS and HTML according to mockup changes.
- Added xanalytics icons to listing page.
- Added xanalytics management command.

8.2.13 Version 0.7.0

- Implemented `Select2` element to refactor `select2` widgets.
- Added checkboxes to allow user to uncheck items in export panel.
- Sped up indexing using caching.
- Made checkbox for `Allow multiple terms` in the taxonomy panel. consistent with the rest of the UI.
- Implemented export of static assets.
- Fixed user menu display on LORE welcome page.

8.2.14 Version 0.6.0

- Modified learningresource panel to include multi select.
- Fixed export button not appearing in certain situations.
- Added test for StaticAsset.loader.
- Added export functionality for learning resources.
- Added select2-bootstrap-theme bower component.
- Added Select2 to the JS libraries.
- Created ICheckbox React component.
- Made XML preview box for a LearningResource should be read only.
- Pinned all versions.
- Avoided hitting the database for the search page.

- Added field to Vocabulary to define if it can contain multiple terms.
- Incremented xbundle version.
- Added test for ManageTaxonomies.loader.
- Changed vocabularies listing page to match the design.
- Fixed broken links in the footer.
- Removed console.error statement.
- Fixed bug where export checkboxes were not updated in sync with export count.
- Fix test failures due to pylint dependency chain.
- Created StatusBox component to hold messages and errors.
- Added shopping cart for export.
- Changed response vocabulary name to match input and avoid key collision.
- Added docker support for running worker or Web process by environment.
- Extended tests for manage_taxonomies.jsx file.
- Added description path to listing page.
- Removed export view which isn&#39;t used anymore.
- Refactored code for reloading module into a function.
- Refactored permission check for listing view.
- Updated Haystack to 2.4.0 - Removed automatic index update from deployment.
- Fixed preview link not showing up in list view.
- Grouped REST tests by common endpoint.
- Changed vocabulary term indexing from string to integer.
- Implemented preview link for learning resource panel.
- Added sorting to search results.
- Implemented learning resource panel updating on every panel open.
- Used different haystack index for tests to prevent conflict with web application.

8.2.15 Version 0.5.0

- Fixed display of vocabulary terms containing spaces.
- Fixed comparison of FileFields to strings.
- Fixed typo in search hint.
- Added bootstrap style to vocabulary learning type checkboxes Closes #337
- Changed search box description.
- Fixed mutating of this.state which is forbidden.
- Added static file parsing to HTML elements.
- Removed vocabulary forms since we are doing this via REST API and React instead.
- Reported code coverage for javascript on the command line.

- Added function to obtain collections.
- Set QUnit timeout to fix test error reporting.
- Added HTML reporting of javascript tests.
- Added panel for static assets.
- Added link to request create repository permission.

8.2.16 Version 0.4.0

- Added view to serve static assets and modified REST API.
- Added fix and test for handling deleted Elasticsearch index.
- Refactored manage_taxonomies.jsx and related tests.
- Sped up test discovery by removing node_modules from search.
- Added learning resource types to manage taxonomies UI.
- Added learning_resource_types API and learning_resource_types field for vocabularies.
- Fixed bug with file path length in static assets.
- Added learning resource UI to edit description and terms.
- **Upgraded several packages**
    - Bootstrap
    - uwsgi
    - static3
    - elasticsearch
    - django-bootstrap
    - django-storages-redux
- Added terms to the readonly lists.
- Allowed blank descriptions for LearningResource model.
- Implemented Enter key to add taxonomy term and added test case to fix coverage.
- Updated Django to 1.8.3
- Correct LORE production URL in Apiary doc.
- Added checkbox styling to vocabulary/term facets.
- Fixed error message on unsupported terms in learning resource.
- Fixed facet checkboxes not showing in production.
- Fixed course/run highlight bug.
- Default checked radio button for Manage Taxonomies -> Add Vocabulary.
- Fixed vertical alignment of taxonomy tabs.
- Fixed error message for duplicate vocabulary.
- Added docker container for javascript testing.
- Added checkboxes and ability to toggle facets.

- Added html coverage report for javascript.
- Added shim configuration to karma test runner.
- Implemented learning_resources API.
- Members REST API docs.
- Linked video transcripts to learning resources.
- Parse static assets from LearningResource.
- Removed unused patterns to limit memory use.
- fix css to make list vertical align.
- Installed JSXHint and configured JSCS to work with JSX files.
- Included JSX files in coverage results.
- Allow only usernames and not emails in the Members add input.
- Added test case, tested menulay all scenarios.
- Moved coverage CLI script to utils directory.
- Fixed buttons alignment problem in members panel.
- Fixed error message behavior for manage taxonomies tab.
- Added ability to filter vocabularies by learning resource type.

8.2.17 Version 0.3.0

- Added UI to add and remove repository members.
- Added form for adding new vocabularies.
- Added manage taxonomies panel and button.
- REST for repo members.
- Implemented taxonomy model delete cascading.
- Renamed “Copy to Clipboard” to “Select XML”
- Setup JSX processing requirements.
- Fixed mis-resolutioned learning resource type icons.
- Converted several large HTML blocks into include files.
- Switched from using main.js for everything to multiple modules.
- Installed lodash.
- Added CSRF jQuery initialization code.

8.2.18 Version 0.2.0

- The search bar performs full-text search over the learning resources in the repository, the search results replace the contents of the listing page.
- Full-text search includes taxonomy facets.
- Learning resources details are displayed in a panel that slides out from the right side of the page.

- Glyphs for learning resources types are displayed in the left side panel for facets.
- LORE's RESTful web service documentation is available. (http://docs.lore.apiary.io)
- Authorizations are in place for taxonomy endpoints in LORE's web service.
- Relationships between learning resources and static assets are captured.
- Roles app has additional features.

Other Changes

- Switched to using get_perms for cleaner code.
- Added JavaScript infrastructure to run unit tests.

8.2.19 Version 0.1.0

- Added taxonomy app with models.
- Added learning resources app.
- Basic Import Functionality
- CAS Integration
- Added forms to taxonomy app.
- Added welcome page.
- Logging support
- Added sphinx documentation project.
- Added add and edit forms for vocabularies.
- Added listing page.
- Added base UI templates.
- Styled listing page.
- Added footer to listing page.
- Added link to repository in repository base template.
- Added support for asynchronous course imports.
- Added rest app with support for RESTful API.
- Added initial authorization support.
- Added login requirement for taxonomy app.
- Switched to using Django storage for course uploads.
- Switched to using Haystack/ElasticSearch for listing page.
- Protected course imports.
- Protected export view.
- Added faceted filtering.
- Added new manage repo users permission.
- Fixed repository listing page to only show results for a single repo.

8.3 LORE Release Workflow

8.3.1 Quick Start

prerequisites:

- `git-release-notes` - https://www.npmjs.com/package/git-release-notes
- `git-flow` - https://github.com/nvie/gitflow
- You've configured Git-Flow to use LORE branch naming scheme. (See *Git-Flow* section below)

These instructions presume you're releasing LORE version 0.10.0.

```
git fetch
git checkout master
git pull
git flow release start 0.10.0
vi lore/settings.py # change version number
git-release-notes v0.9.0..master util/release_notes_rst.ejs
vi RELEASE.rst # paste results of prior command at top of file, updating the release number
git add RELEASE.rst lore/settings.py
git commit -m "Release 0.10.0"
git flow release publish 0.10.0
# Verify that the Travis-ci build succeeds
git-release-notes v0.9.0..master util/release_notes.ejs
# Create PR on the ``release-candidate`` branch named ``rc/0.10.0`` (The
# PR description isn't important)
# Merge this PR immediately to kick off the build
# Create PR on the ``release`` branch named ``rc/0.10.0`` using the
# console output from git-release-notes v0.9.0..master util/release_notes.ejs
# as the PR description
# Once the ``release`` branch deploys on https://lore-rc.herokuapp.com ,
# Instruct developers to view the ``release`` branch PR description and
# validate their commits on https://lore-rc.herokuapp.com
# Wait for all checkboxes to be checked
# Check with Dev-Ops buddy whether migrations need to be run for this release
git flow release finish 0.10.0
git push --tags
# Send release notification to LORE mailing list - odl-lore-announce@mit.edu
```

8.3.2 Detailed Instructions

Create, commit, and push release candidate branch

```
git fetch
git checkout master
git pull
git flow release start 0.10.0
```

Update the `VERSION` number in `lore/settings.py` following Semantic numbering practice.

```
vi lore/settings.py # change version number
```

Generate the *release notes* in reStructuredText format:

```
git-release-notes v0.9.0..master util/release_notes_rst.ejs
```

Update release.rst with this version's release notes (the console output generated by running release_notes_rst.ejs). Edit the new release notes to correct the release number, make each statement in the past tense, capitalize the first word, and end sentences with a period.

```
vi RELEASE.rst
```

Add, commit, and push the two files you have updated to publish the release.

```
git add RELEASE.rst lore/settings.py
git commit -m "Release 0.10.0"
git flow release publish 0.10.0
```

The "release publish" (push) will trigger a build on Travis CI. https://travis-ci.org/mitodl/lore/builds For each commit throughout the release, it's good practice to check that the build succeeded before proceeding.

Create Pull Requests

Developers will start their verification tests as soon as they learn that you've created the PR with the checkboxes. Since the deployment to lore-rc can take a while, merge the release-candidate branch PR as soon as you create it to move its deployment off the critical path.

Make three PRs based on the release branch you just published:

- One to the release-candidate branch named rc/0.10.0
- Merge this PR immediately to kick off the build.
- One to the release branch named rc/0.10.0 containing a checklist of commits
- one to the master branch named Release 0.10.0

Run git-release-notes with the other template to create the Pull Request descriptions for the release branch.

```
git-release-notes v0.9.0..master util/release_notes.ejs
```

and paste the Markdown output as the description of the PR to the release branch rc/0.10.0 (This is the one with the checkboxes.)

Tell developers to test their changes on the LORE release candidate server, http://lore-rc.herokuapp.com and then bug team members until all the check boxes are checked.

Finish the release

When all the boxes are checked finish the release.

```
git flow release finish 0.10.0
git push --tags
```

Follow this by merging the remaining PRs and sending a release notice to LORE users.

The email address of the LORE mailing list we use for release notification is odl-lore-announce@mit.edu It is a mailman list located here: https://mailman.mit.edu:444/mailman/admin/odl-lore-announce/general The list is set to hold all emails for review by list moderators who are notified by email when an email is received.

Release Process Roles

the important piece for the dev-ops person is migrations so the devops person should actually do the merge to release so they can run migrations before they do so which means we are almost exactly following the edX release strategy, where there is a release master and “dev-ops buddy”.

`git-release-notes`

Create the release notes with a utility called `git-release-notes`. https://www.npmjs.com/package/git-release-notes Its output is written to the console.

`git-release-notes` takes two command line arguments. The first argument takes a range of git references to determine which commits to include in the release. In the example given it is all commits from the last release to what is in `master`.

The second command argument specifies a template to format the output. LORE has two templates.

- `util/release_notes_rst.ejs` formats the output as reStructuredText for inclusion in `RELEASE.rst`.
- `util/release_notes.ejs` formats the output as Markdown so as to precede each commit with a checkbox, and follow each with a link to the developer’s commit history. Add these release notes as the description for the GitHub PR to the `release` branch.

Git-Flow

The LORE project uses a different branch naming scheme than is the Git-Flow default. The easiest way to configure this alternate scheme is to add this block to your `~/.gitconfig` file.

```
[gitflow "branch"]
    master = release
    develop = master
[gitflow "prefix"]
    feature = feature/
    release = rc/
    hotfix = hotfix/
    support = support/
    versiontag = v
```

8.4 LORE Test Plan

LORE is known to render differently in Firefox and Chrome, with Chrome producing less satisfactory results. So testing on Chrome is recommended.

8.4.1 Repositories

Create a repository

Note: You may only create a repository if you have the `staff` role in Django.

Action	Result	Notes
On home page, click `Create Repository` link.	The repository listing page appears.	
On `Create Repository` page, enter repository title and description. Both are required. Click `Add Repo`.	The repository listing	

8.4.2 Courses

Import a course

Note: Since many LORE issues are related to large size and "creative" content, import large and potentially problematic courses such as 8.01x ().

A large course import can be slow, anywhere from several minutes to several hours.

Action	Result	Notes
From the listing page, click `Import Course`.		
Click `Browse` to select a course bundled in `tar.gz` format. Once selected, click `Upload Course`.	Course title appears in 'Course' facet list with the total number of LRs following the title.	

8.4.3 Taxonomies

Include occasional non-alphanumeric characters in vocabulary and term names.

Create vocabulary for a range of term types

Action	Result	Notes
Open `Manage Taxonomy` pane and click the `Add Vocabulary` tab.	Enter `Name` and `Description` fields, both are required.	Consider using unicode and punctuation in the names.
Select item types and click `Managed` and leave other choices unchecked. Click `Save`.	The Vocabularies tab appears with the created vocabulary at the bottom of the list. In addition, the vocabulary will appear in the facet panel with the psuedo term `not tagged`.	
Repeat the initial step, this time select item types and click `Tag Style (on the fly)`. Click `Save`.		
Repeat the steps to create `Managed` and `Tag Style` vocabularies, but this time click `Allow multiple terms...` and save as before.		

Add terms

Action	Result	Notes

Assign terms to LRs

Action	Result	Notes
Click a LR to open it’s detail pane and navigate to the `Metadata` tab.	You will see `Vocabularies` and `Terms` dropdown controls.	You must have created a vocabulary containing at least one term to assign terms to LRs.

Edit unassigned terms

Action	Result	Notes

Edit assigned terms

Action	Result	Notes

Delete unassigned terms

Action	Result	Notes

Delete assigned terms

Action	Result	Notes

8.4.4 Vocabularies

Action	Result	Notes

8.4.5 Facets

Action	Result	Notes
Click on twisty to collapse Course facet	Twisty changes direction and course facets are hidden.	

8.4.6 Search

Action	Result	Notes
Enter “momentum” in the search control and click the magnifying glass icon.		

8.4.7 Learning Resources (LR)

Action	Result	Notes
Open the LR pane.	A three-tabbed pane will slide out from the right side of the page.	

8.4.8 Export

Action	Result	Notes
Click multiple LR 'export' links	The arrow on the link becomes a check and a blue button labeled "Export" appears in the upper right corner of the page.	Include one or more of each LR type and include some that have static assets.
Click the big blue 'Export' button	A dialog will appear to download the export file.	
Click "OK" to download the CSV file	What happens when the file is downloaded depends on how your computer configuration. You may see a CSV file, a directory of the file's contents, or the file opened by a spreadsheet app	
Verify that the contents of the file match the LRs and static assets selected for download.	The directory structure of the file contents should have directories for each LR type containing static assets in a subdirectory.	

8.4.9 Sort

To fully test the sort feature, the repository must contain analytics data. A script can be run to load fake analytics data if the analytics service is still unavailable.

Action	Result	Notes
Sort by 'Number of Views (desc)'		
Sort by 'Number of Attempts (desc)'		
Sort by 'Average Grade (desc)'		
Sort by 'Title (asc)'		

8.4.10 Authentication/Authorization

Action	Result	Notes

8.4.11 Members

Add member

Note:

- Members must be known to the site before you can add them. They become known to the site by navigating to the site and authenticating.
- While it appears that LORE validates the email, it does not. Validation is an artifact of the authentication process.

Action	Result	Notes
Add a member as Administrator		
Add a member as Curator		
Add a member as Author		
Delete an Administrator		
Delete the last Administrator for the repository		
Delete a Curator		
Delete an Author		

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Python Module Index