INSURANCE TECHNOLOGIES

Localization Enhancements 2.18 August

FIRELIGHT BASE

FireLight®

Platform

LOCALIZATION ENHANCEMENTS 2.18 AUGUST

Document Version: 1.0

Published: June 29, 2020

Insurance Technologies, LLC

Insurance Technologies, LLC
Two South Cascade Avenue
Colorado Springs, CO 80903
USA
Phone: 719.442.6400
FAX: 719.442.0600
Internet E-Mail: info@insurancetechnologies.com
Website: http://www.insurancetechnologies.com

Table of Contents

Design Approach - Localization Enhancements 2.18 August

Enhancing localization will make it easier for to manage and maintain for both Insurance Technologies and clients.

Impacts:

Localization-Translation (using keys to support translation of data items)

Localization-Language Support (updated keys for existing localized text for enhanced language support/same keys used for each language)

1 Update Localization Key for Ease of Management and Maintenance

1.1 Enhance the Keys used for Localized Text

Currently the key used to identify localized text is based on the text itself, which is long and can change. This key needs to change to a readable and consistent value. So, when the text changes in FireLight the key does not change along with it. This will involve changing the logic in FireLight to request localized text using this key instead of the text itself. It will also include the logic that scans the FireLight code base for Localized text and updates/inserts that data into the database.

The Localized text in FireLight needs to have a timestamp associated with it to support version control. This will allow us to determine what text has changed within a specified time range (i.e. what changed since the last release).

1. Provider's localized text function will also need to be updated.

2. RAZOR code needs updated.

3. All other localized text across system.

4. Backwards compatibility

5. Naming convention of keys is also required to be created.

Acceptance Criteria

- Keys for localized text changed to a readable, consistent value.
- Localized text is tied to keys that can be used in the database.
- Database is updated using logic that scans FireLight code base.
- Localized text has a timestamp associated to support version control.

- Provider's localized text function is updated.
- RAZOR code is updated.
- All localized text across system is updated.
- Backwards compatibility is functioning as expected.
- Naming convention is created for keys.

1.2 Define Keys and Update Code

Everywhere within the FireLight code base that has localized text will need to define a key that can be associated with that text.

Acceptance Criteria

- Existing coding for localized text updated with a defined key.

1.3 Modify tool that scans code for localization entries

Modify power shell script to accept new keys.

Acceptance Criteria

- New keys are accepted.

1.4 Update localization to have matching keys (English/French)

Currently, localization is being used for English/French translation. When updates are made, the existing keys for English need updated to match the French keys and vice versa. As we transition the records to using the new keys, any old records that are no longer being used will be cleaned up/removed.

Acceptance Criteria

- Existing text is updated for English/French translation to new keys that match. The same key is used for the English as for the French and vice versa.

2 Localization Keys and Values for REACT

Localization is currently not available in REACT, however keys and values for localization can be passed down for use. Storage of the localization data and common logic in REACT will provide the developers a way to easily access text for localization to be setup for use in REACT. Once the text and common logic is created, localization will be applied to all REACT components.

2.1 Pass Down Keys and Values for Localization for REACT

Data will be provided in a single request, so that request needs to happen when the REACT code starts up. Because of how the data is stored in the browser, all data needs to be retrieved if the browser does not have it, or the localization text has changed.

Acceptance Criteria

- Localization keys and values can be used in REACT.

2.2 Store Localization Data for use in REACT

Localization data will be stored in Window.localStorage. The localStorage object needs to be in key value form, since the localization data will include more than just keys and values - including locale (i.e. English, Spanish, etc.), organization and last updated.

Localization data will be stored in a JSON object (one JSON blob for each org and locale). When the REACT code starts, it will consume this JSON blob and make it accessible to the rest of the code.

Acceptance Criteria

- Since this is a technical story the acceptance criteria must be verified by a developer.
  - Localization data is stored in Window.localStorage.
  - localStorage object is in key value form and includes locale, organization, last updated and value.
  - Localization data is stored in a JSON object as one JSON blob for each org and locale.
  - When the REACT code starts, the JSON blob is consumed and accessible to the rest of the code.

2.3 Create Common Logic for use in REACT

Common logic for localization in REACT is required for developers to easily access text. This is similar to the LocalizeText (key) that is used in other areas.

Acceptance Criteria

- We will need to prove that localization data from the database is being displayed on a view.
- Verify that a REACT view can display localized data.

2.4 Apply Localization to REACT components

Localization will be applied to all REACT components.

Acceptance Criteria

- Localization is applied to all REACT components.