Multi-paged Ingest Forms

Multi-paged ingest forms are supported in Islandora 7. They allow the user to step though a series of HTML forms which in turn manipulate one or more Fedora Objects; adding data-streams, adding relationships, manipulating metadata. Once every form page has been submitted, Islandora will automatically ingest all the prepared Fedora Objects.

Multi-paged ingest forms have a concept of "steps," such that a steps are a sequential list of actions. Steps must be traversed to completion, at which point the prepared Fedora objects will be ingested. There are currently two types of steps supported by the multi-paged ingest forms:

Form steps: Expected to return a normal single paged Drupal Form.

Callback Steps: Expected to execute a single function and render nothing.

For every page request to the multi-paged ingest form, a single form step is executed. Zero or more callbacks steps may be executed for every page request. Since callback steps don't render anything to the user they are executed consecutively in the order in which they appear.

For example, we have the series of steps below:

callback_1, callback_2, form_1, callback_3, callback_4, form_2, callback_5

1. Before ***form_1***, renders to the user ***callback_1*** and ***callback_2*** will be executed.
2. When ***form_1*** is submitted, ***callback_3***, ***callback_4***, wile be executed before rendering ***form_2***.
3. When ***form_2*** is submitted, ***callback_5*** will be executed,
4. Note that all the steps have executed all the prepared Fedora Objects will be ingested.

Although optional, it's worth noting that typically form/callback steps are expected to also define **undo** functions. These revert the changes they have been made to the prepared objects. The undo functions are called when a user clicks on the previous button in the form and returns to the previous form step.

To support this a module need only declare what steps they wish to add and what functions those steps perform.

Adding Steps

To let islandora know what steps you are providing you must implement the islandora_ingest_steps hook(s):

```
function hook_islandora_ingest_steps(array $form_state);

function hook_CMODEL_islandora_ingest_steps(array $form_state);
```

Which can be implemented by modules to conditionally add new steps. These functions are expected to return an associative array containing the appropriate steps to apply given the current $form_state:

Each step should consist of a unique name mapped to an array of properties which take different parameters based upon type:

Shared properties

- **type:** The type of step. Either "form" or "callback".
- **module:** A module from which we want to load an include file.
- **file:** A file to include (relative to the module's path, including the file's extension).
- **weight:** Steps are sorted by weight. The expected range between -50 to 50. The order is undefined for steps which have the same weight.

Form properties

- **form_id:** The form building function to call to get the form structure for this step.
- **args:** An array of arguments to pass to the form building function, These will be appended after $form, and $form_state.

Callback properties

- **do_function:** An associate array including:
  - **function:** The callback function to be called.
  - **args:** An array of arguments to pass to the callback function.
- **undo_function:** An associate array including:
  - **function:** The callback function to be called to reverse the executed action.
  - **args:** An array of arguments to pass to the callback function.

Forms do not need to describe their **undo.** It's assumed to take the form of:

```
function form_id_undo_submit(array $form, array &$form_state);
```

Example implementation

```
/**
 * Implements hook_islandora_ingest_steps().
 */
function islandora_basic_image_islandora_sp_basic_image_islandora_ingest_steps() {
  return array(
    'islandora_basic_image' => array(
      'weight' => 10,
      'type' => 'form',
      'form_id' => 'islandora_basic_image_image_upload_form',
      'module' => 'islandora_basic_image',
      'file' => 'includes/image_upload.form.inc',
    ),
  );
}
```

Altering Steps

Sometimes you will want to conditionally include/remove steps based on other steps. You can do this via the alter hook(s):

```
function hook_islandora_ingest_steps_alter(array &steps, array $form_state);

function hook_CMODEL_islandora_ingest_steps_alter(array &steps, array $form_state);
```

Example implementation

```
/**
 * Implements hook_islandora_ingest_steps_alter().
 */
function islandora_marcxml_islandora_ingest_steps_alter(array &$steps, array
&$form_state) {
  if (isset($steps['xml_form_builder_metadata_step'])) {
    $metadata_step_storage = islandora_ingest_form_get_step_storage($form_state,
'xml_form_builder_metadata_step');
    if (isset($metadata_step_storage['association']) && $metadata_step_storage
['association']['dsid'] == 'MODS') {
      $steps['islandora_marcxml_upload'] = array(
        'type' => 'form',
        'weight' => 1,
        'form_id' => 'islandora_marcxml_file_form',
        'args' => array(),
        'file' => 'includes/file.form.inc',
        'module' => 'islandora_marcxml',
      );
    }
  }
}
```

Persisting information

As you seen in the previous step, there are some functions (islandora_ingest_form_get_step_storage) that help retrieve and persist information into the form storage. This allows our steps to share information with one another, as well as manipulate the list of prepared Fedora Objects.

There are two levels of storage: **per step** storage and **shared** storage. Shared storage is first populated with the configuration that was passed to islandora_ingest_form. Step storage will often contain custom data related to that particular step, and any submitted values for form steps:

```
function &islandora_ingest_form_get_shared_storage(array &$form_state);

function &islandora_ingest_form_get_step_storage(array &$form_state, $step_id = NULL);
```

And two functions for grabbing the prepared objects, one for grabbing all the objects, and one for grabbing the current object:

```
function &islandora_ingest_form_get_objects(array &$form_state);

function &islandora_ingest_form_get_object(array &$form_state);
```

Example implementation

```
function xml_form_builder_islandora_ingest_steps(array &$form_state) {
  module_load_include('inc', 'xml_form_builder', 'includes/associations');
  $shared_storage = islandora_ingest_form_get_shared_storage($form_state);
  $metadata_step_storage = &islandora_ingest_form_get_step_storage($form_state,
'xml_form_builder_metadata_step');
  $association_step_storage = &islandora_ingest_form_get_step_storage($form_state,
'xml_form_builder_association_step');
  $association_step_storage['models'] = isset($association_step_storage['models']) ?
$association_step_storage['models'] : $shared_storage['models'];
  $associations = xml_form_builder_get_associations(array(), $association_step_storage
['models'], array());
  $metadata_step_storage['association'] = isset($metadata_step_storage['association'])
? $metadata_step_storage['association'] : current($associations);
  $num_associations = count($associations);
  $select_association_step = ($num_associations > 1) ? array(
    'weight' => 0,
    'type' => 'form',
    'form_id' => 'xml_form_builder_select_association_form',
    'module' => 'xml_form_builder',
    'file' => 'includes/select_association.form.inc',
    'args' => array($associations),
      ) : NULL;
  $metadata_step = ($num_associations >= 1) ? array(
    'weight' => 5,
    'type' => 'form',
    'form_id' => 'xml_form_builder_ingest_form',
    'module' => 'xml_form_builder',
    'file' => 'includes/ingest.form.inc',
    'args' => array($metadata_step_storage['association']),
      ) : NULL;
  return array(
    'xml_form_builder_association_step' => $select_association_step,
    'xml_form_builder_metadata_step' => $metadata_step,
  );
}
```