Enrollment Flow Plugins

See Also: Writing Registry Plugins and Plugin Execution

The name of the Plugin should match the format `FooEnroller`.

As of Registry v4.0.0, Enrollment Flow Plugins are instantiated, and as such, the Plugin should implement a `FooEnroller` model and a corresponding controller (`FooEnrollersController`).

1. This model should `belongTo CoEnrollmentFlowWedge`.
2. The controller should extend `SEWController`.
3. When a new Enrollment Flow Wedge is created, a skeletal row in the corresponding `foo_enrollers` table will be created. There is no `add` operation or view required. The skeletal row will point to the parent CO Enrollment Flow Wedge.
4. When an Enrollment Flow Wedge is edited, the entry point to the Plugin will be `foo_enroller/foo_enrollers/edit/#`. This will be called immediately after the parent Enrollment Flow Wedge is created.
5. Note `CoEnrollmentFlowWedge` has a `hasOne` (ie: 1 to 1) relationship with `FooEnroller`.
6. The table `foo_enrollers` should include a foreign key to `co_enrollment_flow_wedges:id`.
    a. Other tables used by the plugin should reference `foo_source:id`.

For all Registry versions, the entry point for Enrollment Flow Plugins is `foo_enroller/foo_enroller_co_petitions/start/coef:#` for the `start` step, and `foo_enroller/foo_enroller_co_petitions/`*`step`*`/#` for all other steps (where # is the relevant CO Petition ID).

The easiest way to implement the Plugin functionality is to extend `CoPetitionsController`. This way, most of the overhead of processing the request will be handled for you, and your plugin need only implement `execute_plugin_`*`step`* for each step you wish to process. (Note the name of each step is camelCased.) Once your plugin is finished, it should return control to the flow by redirecting back to the main flow, using the URL passed in `$onFinish`. The redirect URL is also available in the view variable `$vv_on_finish_url`.

Sample Enroller Plugin

```
// Plugin/FooEnroller/Controller/FooEnrollerCoPetitionsController.php

App::uses('CoPetitionsController', 'Controller');

class FooEnrollerCoPetitionsController extends CoPetitionsController {
  // Class name, used by Cake
  public $name = "FooEnrollerCoPetitions";
  public $uses = array("CoPetition");

  /**
   * Plugin functionality following petitionerAttributes step
   *
   * @param Integer $id CO Petition ID
   * @param Array $onFinish URL, in Cake format
   */

  protected function execute_plugin_petitionerAttributes($id, $onFinish) {
    // Do some work here, then redirect when finished.

    $this->redirect($onFinish);
  }
}
```

Standard MVC rules apply. Note the corresponding Views will match the action name (eg: `petitioner_attributes.ctp`) and not the function name.

As of Registry v4.0.0, the Enrollment Flow Wedge ID (as provided by CoPetitionsController in a view variable) should be passed through any rendered form. The Wedge ID is also available to the FooEnrollerCoPetitionsController via the viewVars (`$this->viewVars['vv_efwid']`).

```
// Plugin/FooEnroller/View/FooEnrollerCoPetitions/petitioner_attributes.ctp

// Pass the Enrollment Flow Wedge ID
print $this->Form->hidden('co_enrollment_flow_wedge_id', array('default' => $vv_efwid));
```

⚠️ If your plugin executes within an unauthenticated flow, you may need to be aware of the authentication token that is passed with the request. This is particularly important if your plugin will display a form and collect additional data. The easiest way to handle this is to check for the token and, if present, insert it into a hidden attribute:

```
// Plugin/FooEnroller/View/FooEnrollerCoPetitions/petitioner_attributes.ctp

// Pass the token if we have one
if(!empty($vv_petition_token)) {
  print $this->Form->hidden('CoPetition.token', array('default' => $vv_petition_token));
}
```

Rendering Within Petitions

As of Registry v4.1.0, Enrollment Flow plugins can inject information into a CO Petition record display (`/registry/co_petitions/view/X`). To do so, the plugin should define a view element in the file `View/Elements/petitionAttributes.ctp`. The element will be passed the following variables:

- `vv_wedge`: The Enrollment Flow Wedge configuration
- `vv_petition`: The Petition record

⚠️ This interface is *Experimental* and may change in a future release.

Known Limitations

Firefox has a hardcoded redirect limit (default: 20) that can be a problem, especially if there are plugins defined and certain steps are skipped (such as approval). To work around it, at the end of each step a redirect is issued to the next step using a meta refresh on a page that is actually delivered. As long as the number of plugins is less than the redirect limit, this will work around the problem. This does suggest a maximum of ~20 enroller plugins may be defined.

http://kb.mozillazine.org/Network.http.redirection-limit