Forms

Contents

Overview

OpenMRS provides **`Form`**, **`Field`** and **`FormField`** objects through the API as well as schema creation in the webapp. FormEntry Module, HTML Form Entry Module and XForms Module all utilize the built-in `Form` infrastructure. OpenMRS also provides API calls to store serializable resources or metadata on any given `Form` instance.

Forms

Fields

Form Fields

Schemas

Form Resources

Terms To Understand

Unknown macro: 'html'

Overview

FormResource objects can be of any Custom Datatype. Only one FormResource will be stored for any combination of *Form* object and *name* (property on the FormResource). In other words, there can only be one *formentry.xslt* FormResource for any given *Form*. If `FormService.saveFormResource()` is asked to save a new *formentry.xslt* FormResource on a Form that already has a *formentry.xslt* FormResource, the FormResource already stored in the database will be purged before saving the new one.

A FormResource is both a CustomValueDescriptor and a SingleCustomValue. This means that any given FormResource not only holds data but also specifies the kind of data in it, similar to how GlobalProperty objects work in OpenMRS 1.9+. Form edit pages provide a tab to manage FormResource objects on that Form.

⚠ TODO: screenshot(s) of Form Resource management

Designing Form Resources

Working with FormResource objects should follow the same methodology as working with GlobalProperty objects. Following is a unit test from `FormServiceTest` to show how this is done, using a basic DateDatatype:

```
@Test
@Verifies(value = "should persist a FormResource", method = "saveFormResource()")
public void saveFormResource_shouldPersistAFormResource() throws Exception {
        Form form = Context.getFormService().getForm(1);
        FormResource resource = new FormResource();
        resource.setForm(form);
        resource.setName("Start Date");
        resource.setDatatypeClassname("org.openmrs.customdatatype.datatype.DateDatatype");
        Date expected = new SimpleDateFormat("yyyy-MM-dd").parse("2011-10-16");
        resource.setValue(expected);

        Context.getFormService().saveFormResource(resource);
        Integer resourceId = resource.getFormResourceId();

        Context.clearSession();

        FormResource actual = Context.getFormService().getFormResource(resourceId);
        Assert.assertNotNull(actual);
        Assert.assertEquals(expected, actual.getValue());
}
```

API Methods

The OpenMRS FormService API must be used to manipulate FormResources. The following API calls are available from **`org.openmrs.api.FormService`**:

```
public FormResource getFormResource(Integer formResourceId);
public FormResource getFormResourceByUuid(String uuid);

/* usages: */
FormResource xslt = getFormResource(1);
FormResource xslt = getFormResourceByUuid("1e67efde-fe91-4802-8f8e-ba34d01cc926");
```

> *Common API retrieval methods for FormResources.*

```
public FormResource getFormResource(Form form, String name);

/* usage: */
FormResource xslt = getFormResource(form, "formentry.xslt");
```

> *Saves a FormResource. The name property of the FormResource must be unique and should refer to a particular service or module claiming ownership of the resource and the use of that resource. Only one resource for each <i>name</i> will ever be saved on a given Form.*

```
public FormResource saveFormResource(FormResource formResource);
```

> *Saves a FormResource. The name property of the FormResource must be unique and should refer to a particular service or module claiming ownership of the resource and the use of that resource. Only one resource for each name will ever be saved on a given Form.*

```
public void purgeFormResource(FormResource formResource);
```

> *Permanently removes a FormResource.*

```
public Collection<FormResource> getFormResourcesForForm(Form form);
```

> *Retrieves all FormResources for a given form.*

```
public Set<String> getFormResourceOwners(Form form);
```

> *Retrieves all unique owners for resources for a given form.*