

Implementing Consumer
Getting Started
Use Cases
Links & Previews
Delegated UI
Notify Customers
Automatic Bugs

Starting the NinaCRM sample application and the Bugzilla adapter.

Open <http://localhost:8181/ninacrm/> in a web browser. You'll see a sample incident:



At the bottom, find the **Related Defects** heading. This is where we show links to related bugs in Bugzilla; the HTML is a simple unordered list:

Implementing Consumer
Getting Started
Use Cases
Links & Previews
Delegated UI
Notify Customers
Automatic Bugs

Showing UI Previews via Dojo Tooltip Widgets

Throughout this, we'll be using the [Dojo JavaScript toolkit](#) to smooth out browser differences and build UI components like buttons and [tooltips](#).

Open the file `index.jsp` in `/src/main/webapp/` and search for `dojo.addOnLoad(addPreviewMouseOverHandlers)`.

Here, when the page is done loading we use the `dojo.query()` method to get all the links on the

Implementing Consumer
Getting Started
Use Cases
Links & Previews
Delegated UI
Notify Customers
Automatic Bugs

Earlier in this tutorial, we walked through an implementation of Delegated UIs for OSLC4JBugzilla, both for [selecting bugs](#) and [creating new bugs](#). In addition to providing the UI and handling the results, the OSLC4JBugzilla adapter (or any other OSLC provider application) announces in its [Service Provider Documents](#) the URL location and recommended size of the UI.

The application that wants to use the Delegated UI (the OSLC Consumer) creates an `<iframe>` for the Delegated UI so that the user can interact with it. The Consumer application must also listen to the `<iframe>` do something with the results of the user's actions.

UI Creation
Factory
Implementing Consumer
Getting Started
Use Cases
Links & Previews
Delegated UI
Notify Customers
Automatic Bugs

work with any OSLC Provider.

As we noted when [we implemented Service Providers and Catalogs](#), one of the cores of OSLC is that clients should not have to hard-code any URLs other than a Service Provider Catalog. Clients should be able to parse the Catalog and navigate from the Catalog to the Service Providers; the Service Providers will then expose the available OSLC services.


If you'd like to follow along with a real Service Provider Catalog or Service Provider, see the [Viewing the machine-readable formats of a Service Provider Catalog](#) section near the bottom of [this section](#).

OSLC for Developers
Tutorials
Services & Design Patterns
Security
Integration Tutorial
OSLC Introduction
Running Examples
Implementing Provider
Getting Started
Planning Provider
Service Resources
Intro to OSLC4J
UI Preview
UI Selection
UI Creation
Factory
Implementing Consumer
Getting Started
Use Cases
Delegated UI
Notify Customers
Automatic Bugs

Results

With all these in place, you should now be able to add links to our incident pages with delegated OSLC dialogs.

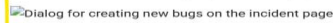
Here's the dialog for selection:



You can search for bugs directly:



You can also create new bugs right from the page:



Either way, the new or selected bug appears on the page (complete with the ability to see a UI preview of the bug):



At this point, we've completed our first milestone goals: the CRM system now uses links, OSLC UI Preview, and OSLC delegated dialogs to make it faster for support reps to find and create bugs.

Next: Part 2.4, Notify Customers

Table of contents

Introduction to OSLC Delegated UI
The mechanics of Delegated UI
Parsing the Service Provider Documents
Adding Delegated UI dialogs to the NinaCRM
[Buttons to launch the dialogs](#)
Results

Implementing Consumer
Getting Started
Use Cases
Links & Previews
Delegated UI
Notify Customers
Automatic Bugs

Using a Service Provider Catalog to find a Service Provider

For our OSLC-CM Bugzilla Adapter (or any OSLC provider), the starting point for exploring OSLC capabilities is the [Service Provider Catalog document](#).

You can read more about [Implementing Service Provider Catalogs for our Bugzilla Adapter](#) [here](#).

In short, we represent every Bugzilla Product as a [Service Provider resource](#), and we collect all of those Service Providers in one Service Provider Catalog.

The general principle is that clients should only need to know the URL for the Catalog; from the

Implementing Consumer
Getting Started
Use Cases
Links & Previews
Delegated UI
Notify Customers
Automatic Bugs

```
</oslc:Service>
</oslc:service>
</oslc:ServiceProvider>
</rdf:RDF>
```

You can read more about [implementing Service Providers](#) and [implementing creation factories](#) for our Bugzilla adapter.

Of most interest to our team developing a way to automatically create bugs are the contents of

Implementing Consumer

Getting Started

Use Cases

Links & Previews

Delegated UI

Notify Customers

Automatic Bugs

newly created bug.

Try it out!

If you'd like to more details or want to try to post a bug using RDF/XML, see [our walkthrough of Implementing a Creation Factory for our Bugzilla adapter.](#)

🕒 June 14, 2025

🕒 May 23, 2019

Eclipse Lyo

Eclipse Lyo

Lyo Java SDK

Overview

Setup Guide

Eclipse Setup

Migration Guides

Migration Overview

Lyo 2.x → 4.x

Lyo 4.x → 5.x

Lyo TRS support

Additional components

See [Sample applications and code](#) for example applications that are based on Eclipse Lyo. Specifically, check out [OSLC Open Project Reference Implementation](#) to see how OSLC works directly with working samples and with a simple server to test against.

Lyo SDK

Lyo's central component is the SDK (Software Development Kit) that helps build REST-based

Table of contents

Table of Contents

Lyo SDK

Lyo Designer

Lyo TRS support

Additional components

Next will share soon...