

Dynamic & Dependency Plugin Properties Options

[Plugin Properties Options](#) allowed us to provide an UI for user to configure a plugin. This tutorial is to demonstrate how we can make use of Element (Plugin) Select property type "[elementselect](#)" to achieve a dynamic and dependency configuration page.

In this tutorial, I will create a sample process tool which will print out all the values of the dependency fields in that plugin. Please download the source code and sample plugin KB [here](#).

As usual, we need a JSON file for plugin properties options in our plugin. My sample plugin only has 1 "[elementselect](#)" field named "option" and the url pointed to its own [web service implementation](#) to retrieve dependency properties options.

```
File : /properties/app/pluginOptions.json
[
  {
    title : 'Edit Dependency Plugin Options Sample Tool',
    properties : [
      {
        name:'option',
        label:'Option',
        type : 'elementselect',
        options : [
          {value : '', label : ''},
          {value : 'option1', label : 'Option 1'},
          {value : 'option2', label : 'Option 2'}
        ],
        required : 'true',
        url : '[CONTEXT_PATH]/web/json/app[APP_PATH]/plugin/org.joget.sample.DependencyPluginOptions
/service?action=getJson',
      }
    ]
  }
]
```

And then, create another 2 JSON files for properties options which dependent on value "option1" and "option2" as below:

```
File : /properties/app/pluginOptions_option1.json
[
  {
    title : 'Option 1',
    properties : [
      {
        name:'field1',
        label:'field 1',
        type : 'textfield'
      },
      {
        name:'field2',
        label:'field 2',
        type : 'textfield'
      },
      {
        name:'field3',
        label:'field 3',
        type : 'textfield'
      }
    ]
  }
]
```

File : /properties/app/pluginOptions_option2.json

```
[
  {
    title : 'Option 2',
    properties : [
      {
        name:'field4',
        label:'field 4',
        type : 'textfield'
      },
      {
        name:'field5',
        label:'field 5',
        type : 'textfield'
      },
      {
        name:'field6',
        label:'field 6',
        type : 'textfield'
      }
    ]
  }
]
```

Once we done the properties options JSON definition files, we need to [implement the PluginWebSupport](#) interface to return the dependency properties options based on the selected value.

```
public void webService(HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException {
    boolean isAdmin = WorkflowUtil.isCurrentUserInRole(WorkflowUserManager.ROLE_ADMIN);
    if (!isAdmin) {
        response.sendError(HttpServletResponse.SC_UNAUTHORIZED);
        return;
    }

    String action = request.getParameter("action");
    if ("getJson".equals(action)) {
        String value = request.getParameter("value");

        String output = "";
        if (value != null && !value.isEmpty()) {
            output = AppUtil.readPluginResource(getClass().getName(), "/properties/app/pluginOptions_"+value+".
json", null, true, null);
        }
        response.getWriter().write(output);
    } else {
        response.setStatus(HttpServletResponse.SC_NO_CONTENT);
    }
}
```

In my sample Process Tool plugin execute method, retrieve and print out the dependency options value.

```

public Object execute(Map properties) {
    //To retrieve the values of dependency options object
    Object optionObj = properties.get("option");
    if (optionObj instanceof Map) {
        Map temp = (Map) optionObj;

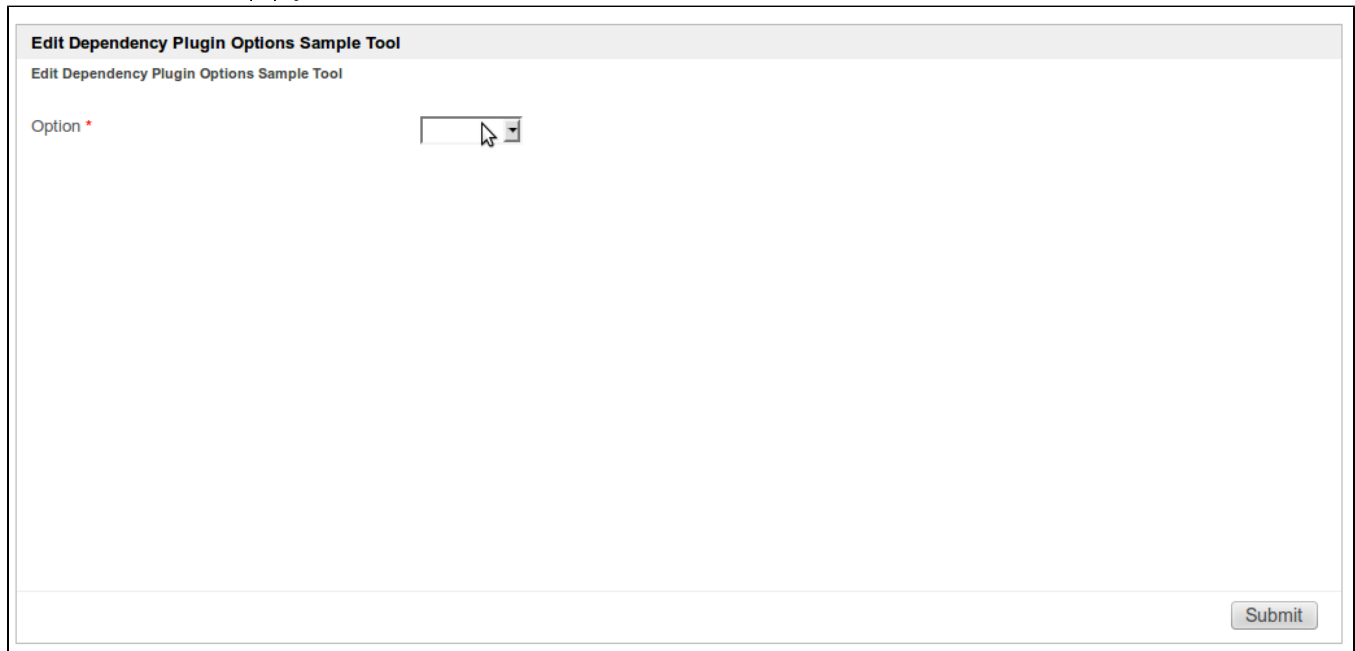
        // to get the value of option
        String option = temp.get("className").toString();
        System.out.println("Option : " + option);

        // to get the value of dependency fields
        Map optionProp = (Map) temp.get("properties");
        for (String key : (Set<String>) optionProp.keySet()) {
            System.out.println(key + " : " + optionProp.get(key));
        }
    }

    return null;
}

```

Below are some screenshots of the sample plugin.



Edit Dependency Plugin Options Sample Tool

Edit Dependency Plugin Options Sample Tool

Option *

Submit

Plugin properties configuration screen when Option is empty.

Edit Dependency Plugin Options Sample Tool

Edit Dependency Plugin Options Sample Tool > [Option \(Option 1\)](#)

Option *

Option 1

< Prev

Next >

Submit

When Option 1 is selected, a new properties page is shown.

Option 1

Edit Dependency Plugin Options Sample Tool > Option 1

field 1

field 2

field 3

< Prev

Next >

Submit

Dependency properties options based on Option 1.

Option 2

Edit Dependency Plugin Options Sample Tool > **Option 2**

field 4

field 5

field 6

< Prev

Next >

Submit

Dependency properties options based on Option 2.

```
Option : option1
field3 : 789
field2 : 456
field1 : 123
```

Plugin output when Option 1 is selected.

```
Option : option2
field5 : def
field4 : abc
field6 : ghi
```

Plugin output when Option 2 is selected.