Set the value of an MLCS custom field using ScriptRunner

ScriptRunner users can easily setup a scripted action which sets the value of a MLCS custom field.

For instance, we can write a scripted workflow postfunction as follows:

Let us suppose that we have a MLCS custom field with name "Place", and that we want our postfunction to set its value on the option with ID 10021 (the script will work regardless the depth of the option, it will find out about all the ancestor options).

Our groovy script will be

```
import com.atlassian.jira.component.ComponentAccessor
import com.atlassian.jira.event.type.EventDispatchOption
import com.atlassian.jira.issue.CustomFieldManager
import com.atlassian.jira.issue.MutableIssue
import com.atlassian.jira.issue.ModifiedValue
import com.atlassian.jira.issue.util.DefaultIssueChangeHolder
import com.atlassian.jira.issue.customfields.option.Option
import com.atlassian.jira.issue.fields.CustomField

CustomFieldManager customFieldManager = ComponentAccessor.getCustomFieldManager();

CustomField customField = customFieldManager.getCustomFieldObjectByName("Place");

def cfVal = issue.getCustomFieldValue(customField);

Option targetOption = ComponentAccessor.getOptionsManager().getOptions(customField.getRelevantConfig(issue)).
getOptionById(10021L);
List<Option> optionsChain = new ArrayList<Option>();
// We store the target option in the array optionsChain, then we add all of its ancestors
optionsChain.add(targetOption);
Option parentOption = targetOption.getParentOption();
while (parentOption != null){
    optionsChain.add(parentOption); parentOption = parentOption.getParentOption();
}

// We need to reverse the order of the options in the array so that the root-level one is the first
Collections.reverse(optionsChain);

customField.updateValue(null, issue, new ModifiedValue(cfVal, optionsChain), new DefaultIssueChangeHolder());
```