



Dynamic Security

Take the protection of your information
to another level.

Kaldeera Dynamic Security 2010

User's guide

Index

Kaldeera Dynamic Security	3
Using Kaldeera Dynamic Security	4
Accessing settings page	4
Enabling or disabling Kaldeera Dynamic Security functionality	6
Configuring Kaldeera Dynamic Security for a list	7
Rules	7
Creating the xml code.....	8
Adding rules	9
Rule order	9
Adding conditions to a rule.....	10
Permissions.....	15
Cascading security	17
Saving settings	18
Reapply security on save	18

Kaldeera Dynamic Security

Kaldeera Dynamic Security (KDS) is a tool that allows you to **apply security to SharePoint® list items and documents**. You can dynamically apply security to your list items or files through a series of rules. These rules depend on the values of the item attributes.

With KDS you will achieve the results you expect **without additional development costs** through its technology based on the application of rules. No need to buy multiple solutions and components. Get **all the functionality in one product!**



Custom security

Modify SharePoint security as you needed is now possible. Immediate results!



No .Net knowledge required

Reduced cost of development. Ability to combine and reuse existing systems.



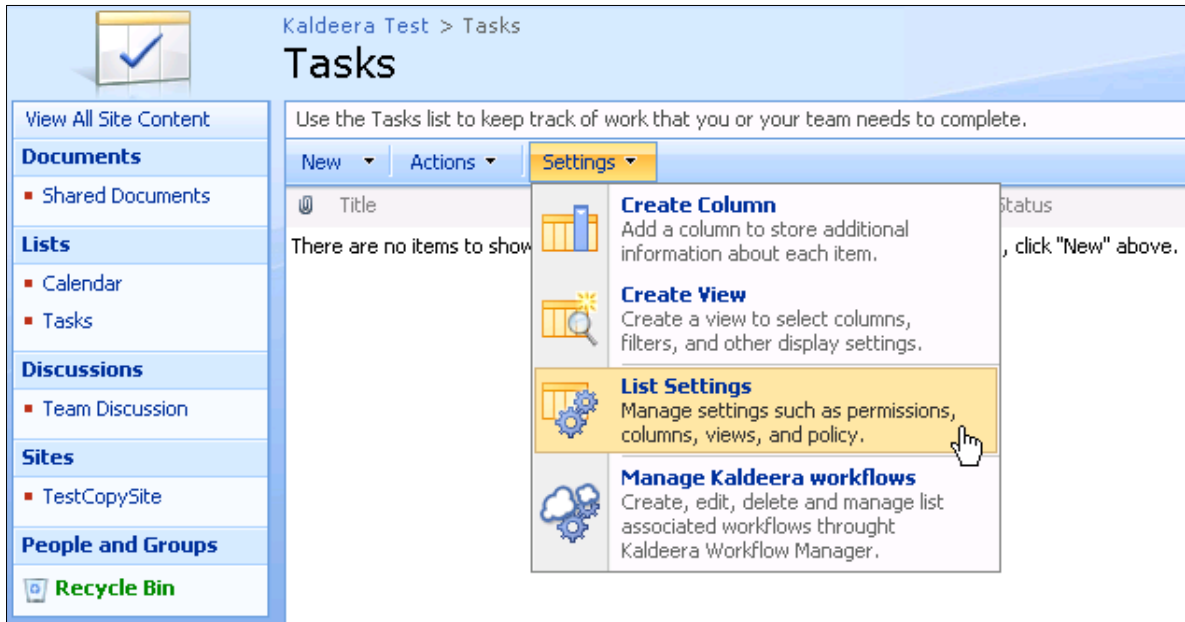
Easy to use

Ideal tool for business users

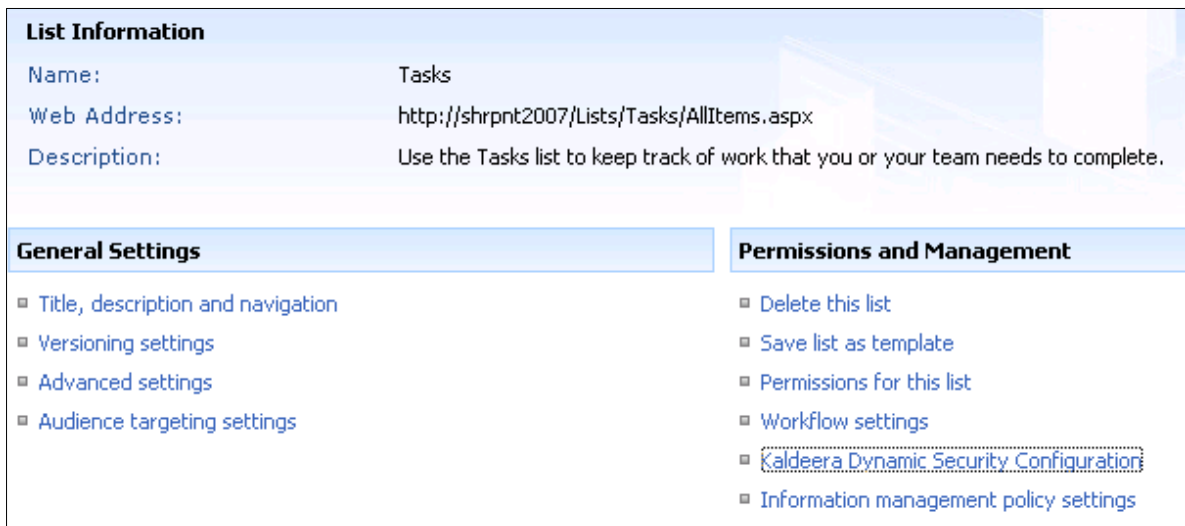
Using Kaldeera Dynamic Security

Accessing settings page

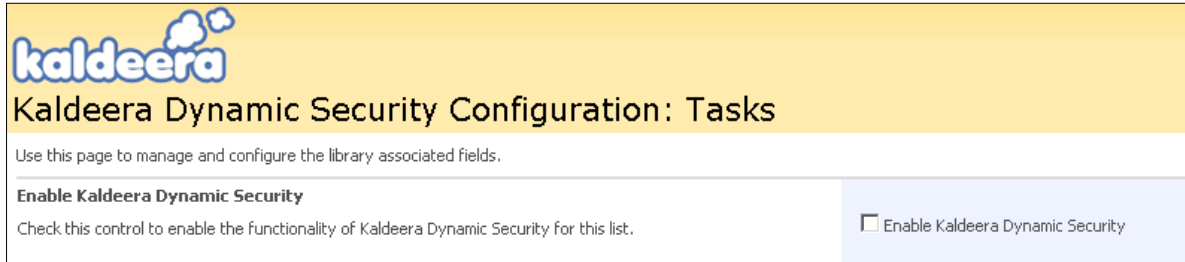
First navigate to a list or document library and click “Settings”.



Select “List Setup”. The list setup page will load. Click on “Kaldeera Dynamic Security Configuration” link under column “General settings”.



Once the link has been selected, Kaldeera Dynamic Security will load.



kaldeera

Kaldeera Dynamic Security Configuration: Tasks

Use this page to manage and configure the library associated fields.

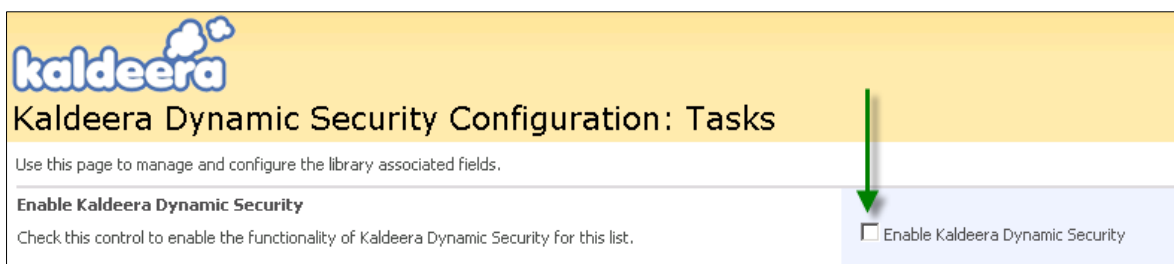
Enable Kaldeera Dynamic Security

Check this control to enable the functionality of Kaldeera Dynamic Security for this list.

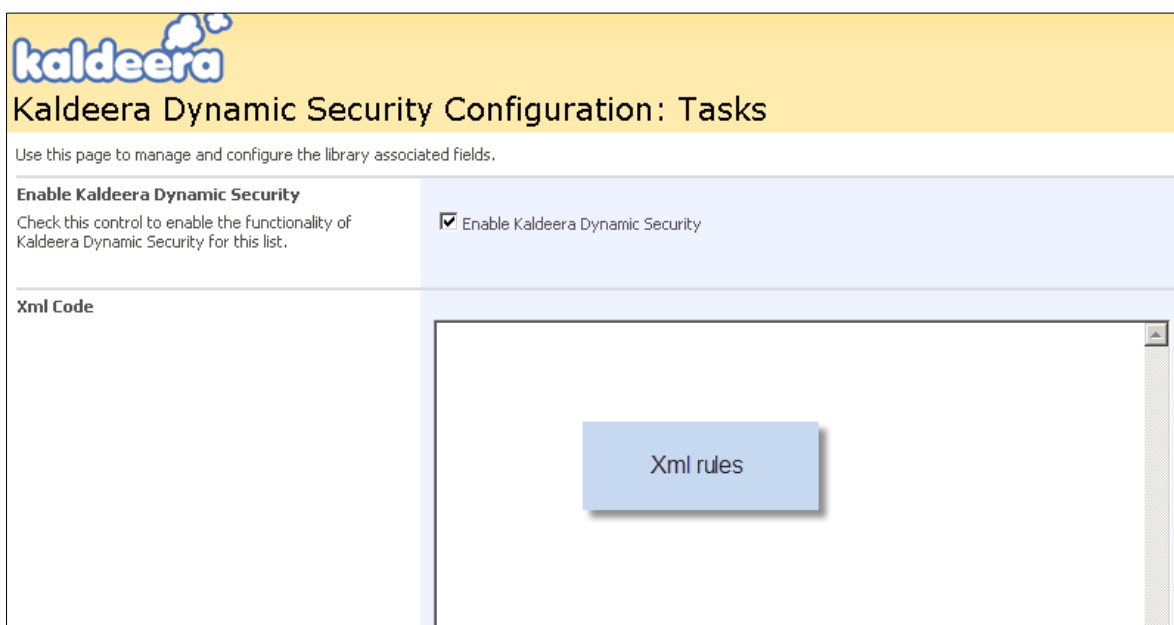
Enable Kaldeera Dynamic Security

Enabling or disabling Kaldeera Dynamic Security functionality

Click on the “Enable Kaldeera Dynamic Security” Checkbox to enable Kaldeera Dynamic Security functionality on a list or documents library.



Once enabled the “Xml Code” textbox control will allow you to write the security rules that will to the list items.



To disable Kaldeera Dynamic Security functionality click again on the “Enable Kaldeera Dynamic Security” Checkbox.

Configuring Kaldeera Dynamic Security for a list

Rules

With Kaldeera Dynamic Security items rights are applied to the list items according to a group of rules set by the administrator.

For example:

- *If the item has “% complete” field set as lower than 50 then set to the SharePoint group “task Managers” collaboration permissions in the item.*
- *If the item has “Project Status” field set as “closed” then delete all access permissions except the read one.*
- *On the rest of cases set to all the users read access.*

Each rule has a condition that is checked when the list item is created/modified.

The condition result can only be true if the condition is met, otherwise the result will be false.

If the rule condition is evaluated and the result is true, the permissions configured for that rule will be applied to the list item. In other case, the next rule will be evaluated.

Kaldeera Dynamic Security assesses the conditions established for each rule from top to bottom. If the rule condition is met (or if all conditions are met because there are several of them), the established permission for that rule will be applied. The rest of the rules won't be evaluated.

Creating the xml code

The xml code to configure Kaldeera Advanced Forms always starts with the root node *Kaldeera.DynamicSecurityDefinition*:

```
<Kaldeera.DynamicSecurityDefinition>  
  xmlCode  
</Kaldeera.DynamicSecurityDefinition>
```

For each rule we want to add we need to add an *If* node:

```
<Kaldeera.DynamicSecurityDefinition>  
  <If test=" ">  
    permissions  
  </If>  
</Kaldeera.DynamicSecurityDefinition>
```

In the test attribute of the *If* node we'll configure the condition(s) for that rule. See Conditions section.

The permissions we want to apply to the rule if the condition is met are allocated inside the *If* node. Each permission needs a permission node:

```
<Kaldeera.DynamicSecurityDefinition>  
  <If test=" ">  
    <permission type="add" member="Item[Author]" permissionName="Read" />  
    <permission type="add" member="Site Owners" permissionName="Contribute" />  
  </If>  
</Kaldeera.DynamicSecurityDefinition>
```

Adding rules

When configuring a list of permissions to be applied we have to define at least one rule.

A rule is a group of permissions that will apply if the conditions configured for that rule are met.

We can configure several rules with different conditions to add different permissions.

Each rule needs an *If* node in the configuration xml:

```
<Kaldeera.DynamicSecurityDefinition>
  <If test="Condition 1">
    permissions A
  </If>
  <If test="Condition 2">
    permissions B
  </If>
  <If test="Condition 3">
    permissions C
  </If>
</Kaldeera.DynamicSecurityDefinition>
```

Rule order

The rule order is important because if the conditions of a rule are met the rest of the rules are not evaluated. If a user meets the conditions of several rules, only the permissions of the first one whose conditions are met will be applied.

Adding conditions to a rule

The rule condition is set into the *test* attribute of the rule tag (the *if* tag).

```
<If test="Condition 1">
```

The conditions are comparisons.

A condition has two parameters or operands and one operator.

For example:

- A = B
- A > 3

“A,” “B” and “3” are parameters or operands.

“=,” “>” are operators.

The condition result can only be true if the condition is met, otherwise the result will be false.

If the evaluation of the condition is true, the permission established by that rule will be applied, or in case several conditions are established for that rule the following condition will be evaluated.

The condition syntax is:

```
Item[FieldName]='text'
```

In this example we are comparing the value of the field named “*FieldName*” with a text.

We can add more than one condition to a rule separating the conditions with the operators AND and OR. The syntax is:

```
Item[FieldName1]='text1' $AND$ Item[FieldName2]='text2' $OR$ Item[FieldName3]=  
Item[FieldName4]
```

If the field is a lookup field (a field that is pointing to another item field in another list), we can select the fields of the item in the other list with the syntax:

```
Item[LookupFieldName][FieldName]
```

Operators of a condition

The possible operators of a condition are:

- =
- !=
- <
- >
- <=
- >=
- Starts with
- Not starts with
- Ends with
- Not ends with
- Contains
- Not contains

The operators <, >, <= and >= are exclusively for math and date comparisons.

The operators “Starts with”, “Not starts with”, “Ends with”, “Not ends with”, “Contains” y “Not contains” are exclusively for text strings comparisons.

Conditions examples

- Example 1:

`Item[Created]='kaldeera\jlopez'`

Type of first operand: "Item field".

Type of second operand: "Literal".

Operator: "="

This condition is met if the author of the item (the user which created the item) is 'kaldeera\jlopez'.

- Example 2:

`Item[PercentComplete]>'40'`

Type of first operand: "Item field".

Type of second operand: "Literal".

Operator: ">"

This condition is met if the field "% completed" of the item is higher than 40.

- Example 3:

`Item[Created]=Item[Responsible]`

Type of first operand: "Item field".

Type of second operand: "Item field".

Operator: "="

This condition is met if the author of the item (the user which created the item) is the user established in the "Responsible" field.

- Example 4:

`Item[Company][Title]='Kaldeera'`

Type of first operand: "Item field".

Type of second operand: "Literal".

Operator: "="

This condition is met if the field "Title" which is in a list to which the field "Company" points to it is equal to "Kaldeera"

Advanced options for conditions

You can establish the condition to be case sensitive or not when comparing operands of a string.

By default, Kaldeera Dynamic Security is not case sensitive when comparing strings.

For example:

David Smith = david smith

To enable the differentiation between upper and lower cases, add the attribute “CaseSensitive” to the rule (The *if* tag). Example:

```
<If test="Item[Author]='david smith'" CaseSensitive="True">
```

In this case:

- David Smith != david smith
- David Smith = David Smith

If you add the attribute “CaseSensitive” for comparisons different to the string type, this will have no effect over the assessment of the condition.

For testing purposes you can disable a rule by adding the attribute *disabled* to “true”.

```
<If test="Item[Author]='david smith'" disabled="True">
```

In this case, the rule won't be evaluated, as if not exists.

Permissions

Once a rule and its conditions are created, we have to set the permissions to be applied to the item in the event the conditions are met.

To do this, add the tag permission inside the rule tag.

```
<If test="Item[Status]='Closed' " CaseSensitive="False">
  <permission type="add" member="Item[Author]" permissionName="Read" />
  <permission type="add" member="Site Owners" permissionName="Contribute" />
</If>
```

The attributes of a permission node are:

Type

The *type* attribute establishes the action to perform.

For now, the only possible action is:

- **Add:** Adds the permission to the item permission list, if not exists.

Member

The user or group to assign permissions. This attribute can be an item field.

permissionName

The permission level to be assigned. In case you want to apply more than one permission add them separated by commas.

Permissions Examples

- Example 1:

```
<permission type="add" member="Item[Author]" permissionName="Read" />
```

Permission type: Add
Permission people or group: Item[Author]
Permission name: Read

This permission will apply to the user who created the item (*Author*) the permission level named *Read*.

- Example 2:

```
<permission type="add" member="Site Owners" permissionName="Contribute" />
```

Permission type: Add
Permission people or group: Site Owners
Permission name: Contribute

This permission will apply to the group “*Site Owners*” the permission level named *Contribute*.

- Example 3:

```
<permission type="remove" member="kaldeera\john" permissionName="Read, Contribute" />
```

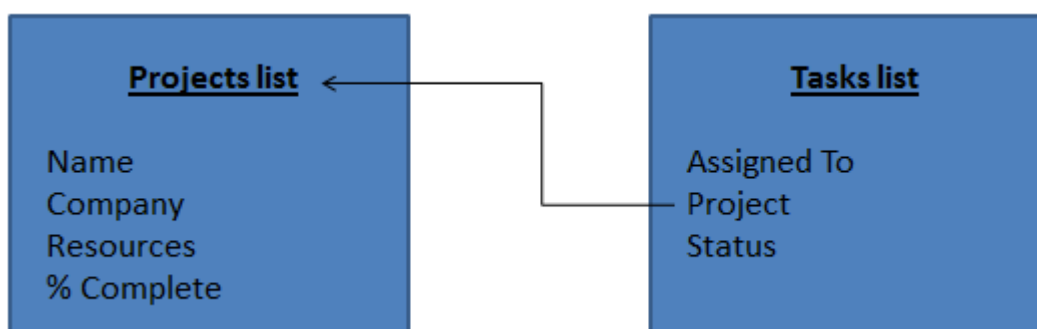
Permission type: Remove
Permission people or group: kaldeera\john
Permission name: Read, Contribute

This permission will remove to the user “*kaldeera\john*” the permission levels named *Read* and *Contribute*.

Cascading security

In some cases we need the capacity to trigger the Dynamic Security in another list whose items are related to our current item (lookup).

For example, we have two lists. The projects list that saves the data of our different company projects and the tasks list that assigns tasks to our employees. The task list is related to the project list with a lookup field.



We want that if a project is completed (*the %Complete field is set to 100%*) all the tasks in the task list related to our project are set to “Read only”.

To accomplish this we need the tag *list*. The syntax is:

```
<list site="Kaldeerasite" list="Tasks" field="Project" />
```

When we modify one project in the projects list this line will trigger the Dynamic Security permission update in all the task list items whose field “Project” is pointing to the project have modified.

The attributes of this tag are:

site

The site name where the list is.

list

The name of the list.

field

The name of the lookup field which is pointing to the other list.


We can add the next configuration xml to the projects list to change the security in the task list:

```
<Kaldeera.DynamicSecurityDefinition>
  <list site="Kaldeerasite" list="Tasks" field="Project" />
</Kaldeera.DynamicSecurityDefinition>
```

Now we can add the next configuration xml to the tasks list to change the security when a project % Complete is set to 100:

```
<Kaldeera.DynamicSecurityDefinition>
  <If test="Item[Project][%Complete]=100">
    <permission type="add" member="All users" permissionName="Read" />
  </If>
</Kaldeera.DynamicSecurityDefinition>
```

Saving settings

When you click on the  button, the configuration inserted in the xml textbox will be saved.

Note: The new configuration will only affect to the new items or when you modify an existent item. If you want to apply the new configuration to the existing items without modifying them you need to check the option “reapply security on save”.

Reapply security on save

If the checkbox “reapply security on save” is selected when the Dynamic Security configuration is saved, the new configuration will be applied to all the items/files in the list/library.

