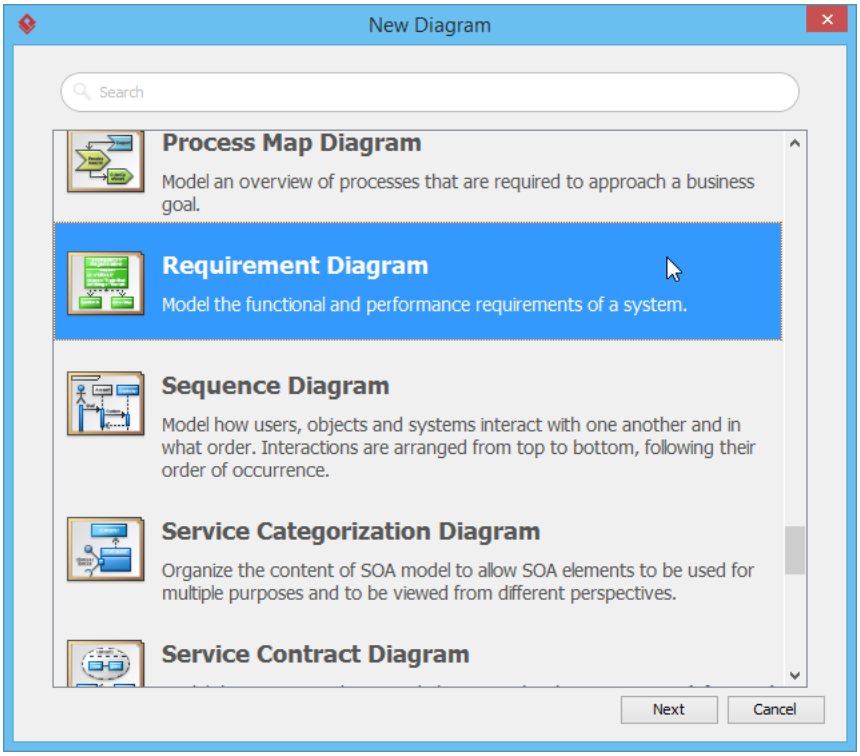
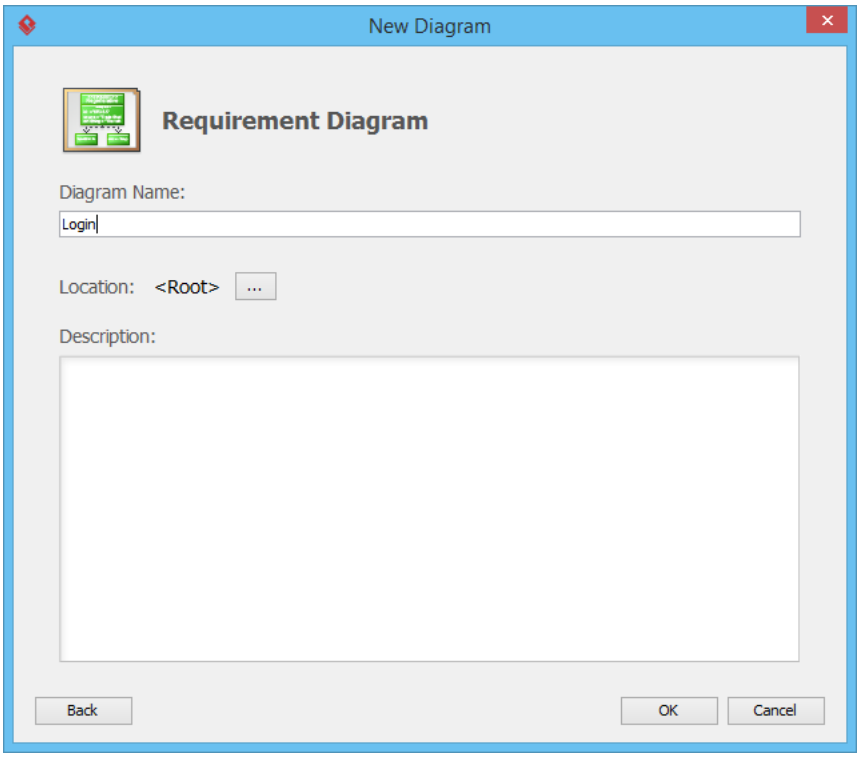
How to Customize SysML Requirement Types?

Written Date : February 4, 2016

1. Create a new project by selecting **Project > New** from the application toolbar.
2. In the **New Project** window, enter *Tutorial* as the project name and click **Create Blank Project**.
3. We will create several requirements in this tutorial. Let's create a requirement diagram first. To create a requirement diagram, select **Diagram > New** from the application toolbar.
4. In the **New Diagram** window, select **Requirement Diagram** and click **Next**.



5. Enter *Login* as the diagram name and click **OK**.

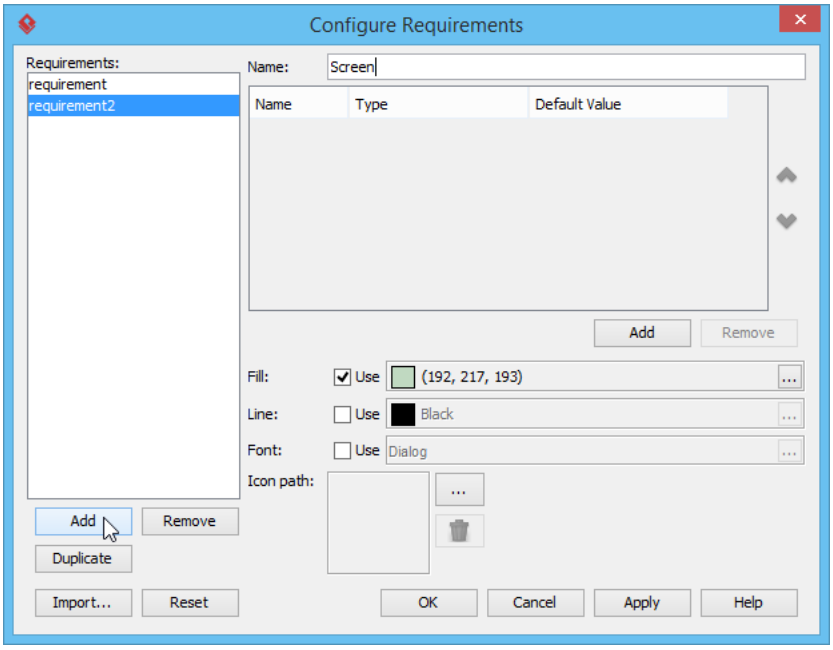


6. On the left side of the diagram, you can see the diagram toolbar, where you can select a tool like a **Requirement**, a **Model**, or a **Test Case** and click on the diagram to create a shape. Normally, you would use the **Requirement** tool to create requirements. But in this tutorial, we will define our own types first. Select **Window > Configuration > Configure Requirements...** from the application toolbar.

7. We will create two requirement types:

Type	Description
Screen	A type of requirement that consists of properties related to a screen design.
Checking	A type of requirement that consists of rules for validating input and the proper response to bad input.

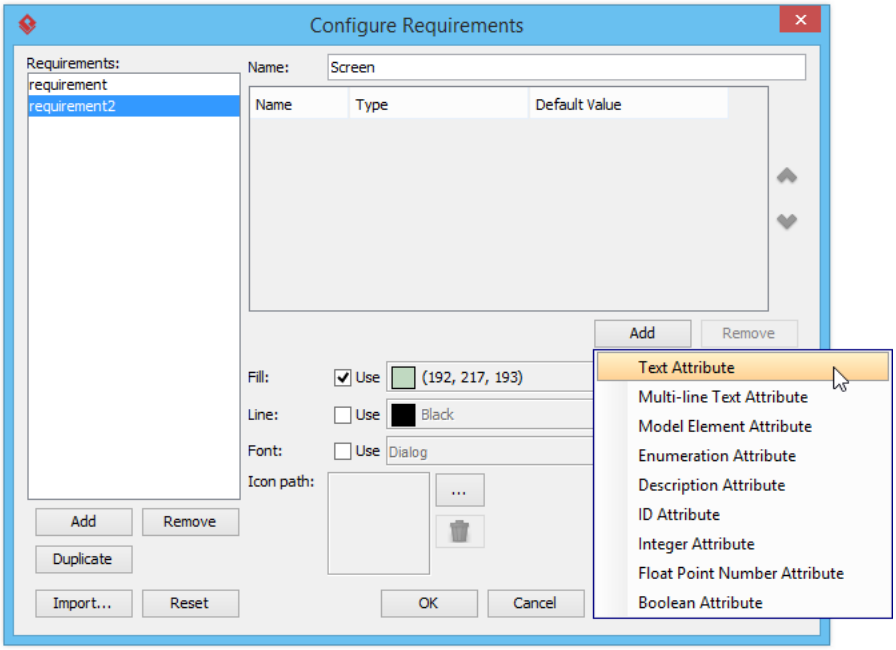
Create the *Screen* type first. Click **Add** at the bottom left of the **Configure Requirements** window. Then, enter *Screen* as the name.



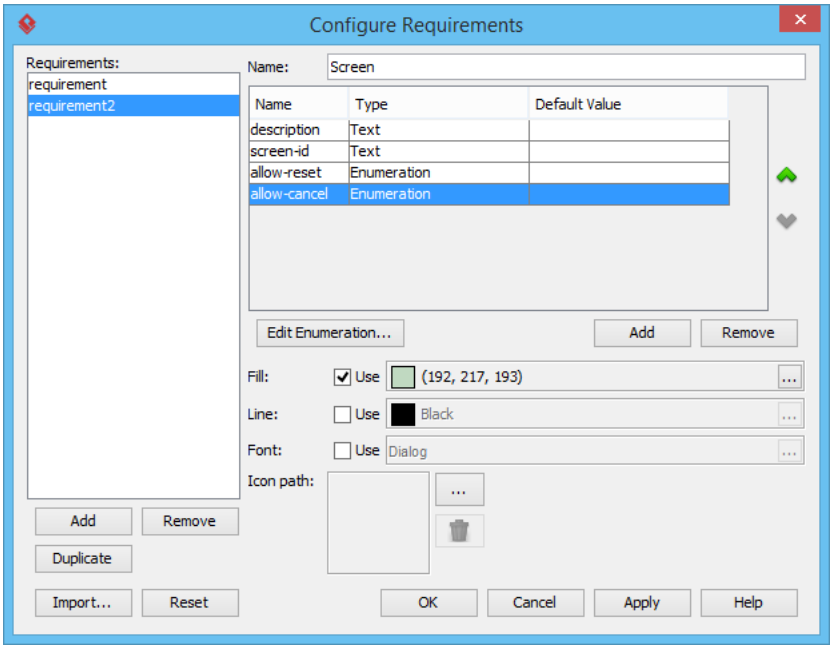
8. What makes the *Screen* type meaningful are its unique attributes (i.e., properties). For the *Screen* type, the following attributes are needed:

Attribute	Type	Description
description	Text	A description of the screen being documented.
screen-id	Text	A unique and internal value for identifying the screen being documented.
allow-reset	Enumeration	Determines whether there is a "Reset" button for clearing fields on the screen.
allow-cancel	Enumeration	Determines whether there is a "Cancel" button to close the screen without proceeding.

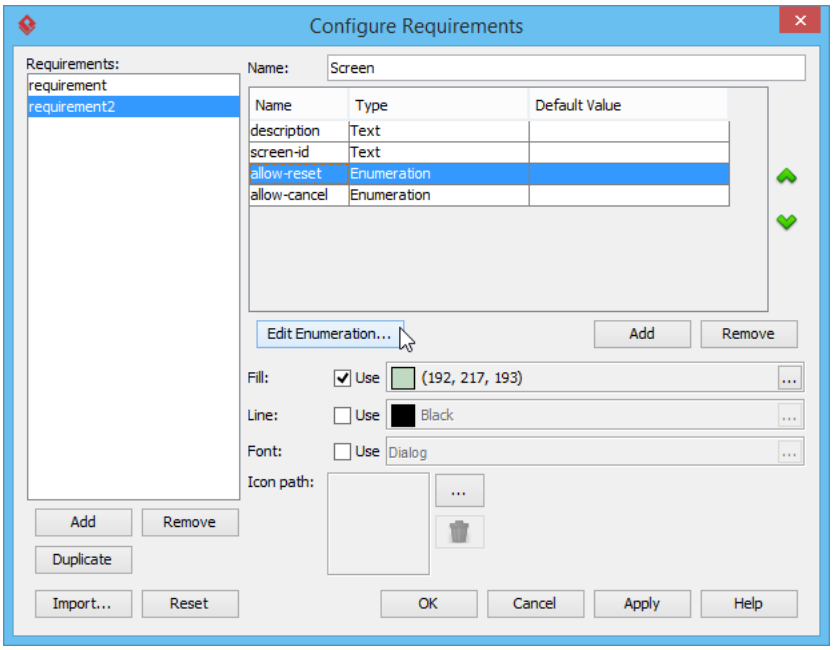
We need to define these properties. Click **Add** on the right side of the dialog box and select **Text Attribute** from the popup menu.



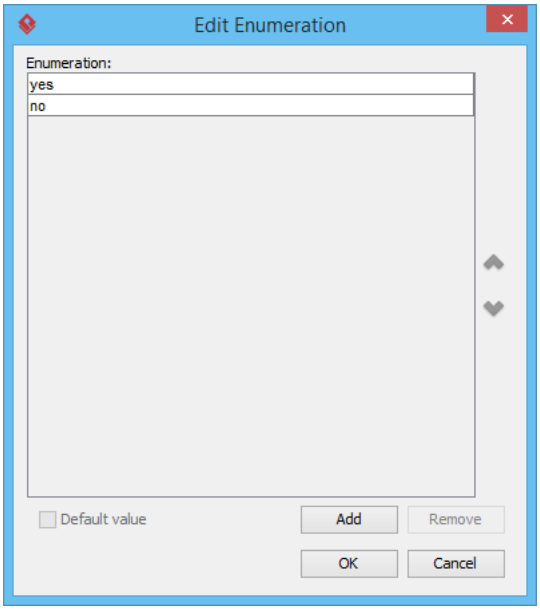
9. Enter *description* as the name. Repeatedly create the remaining properties. Make sure you have chosen **Enumeration** as the type for *allow-reset* and *allow-cancel*.



10. The third attribute, "allow-reset," is an enumeration attribute that enables the selection of "yes" or "no." Select the attribute and click **Edit Enumeration...**.

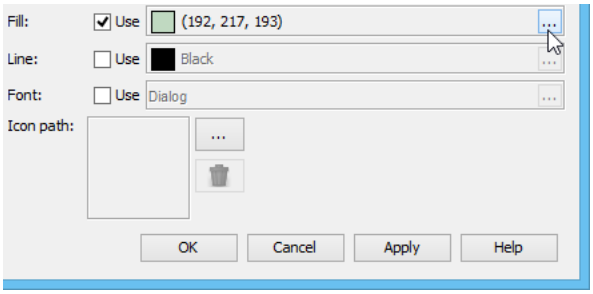


11. In the **Edit Enumeration** window, click **Add** and enter *yes*. Click **Add** again and enter *no*. This creates the two allowed values, "yes" or "no," for this attribute. Click **OK** to go back to the requirement configuration.

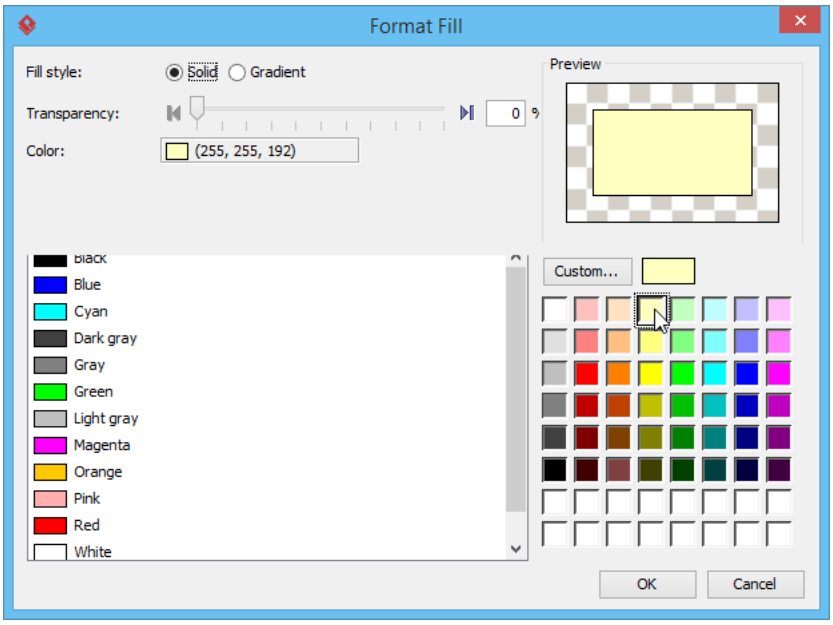


12. Similarly, add the enumeration values *yes* and *no* for the *allow-cancel* attribute.

13. All four attributes have been added. Besides defining requirement attributes, you can also set formatting properties like fill, line, and font styles. These settings will affect how this type of requirement will look in a diagram. Let's try setting a different fill color. Click the **...** button for **Fill**.



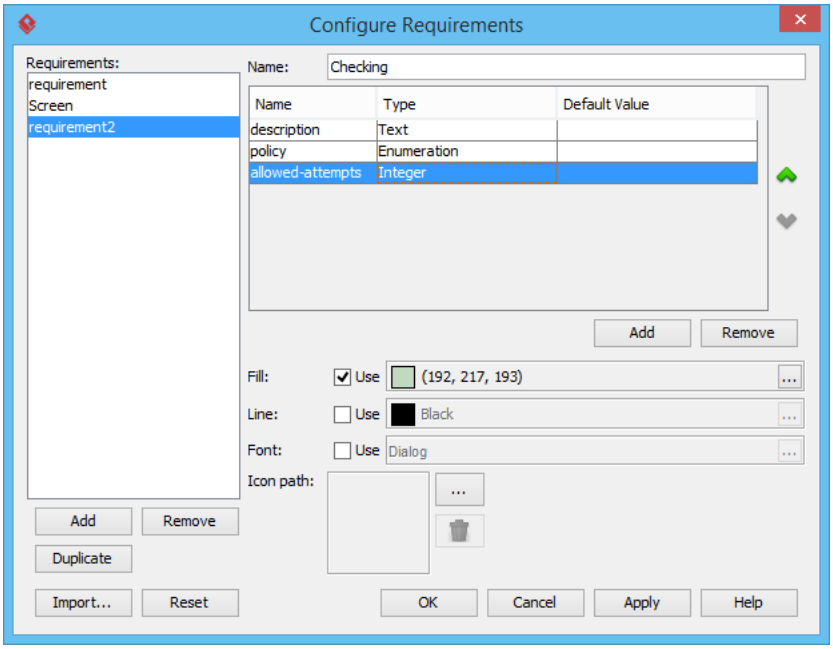
14. In the **Format Fill** window, select yellow and click **OK**.



15. This ends the configuration of the *Screen* type. Apply the same technique to create another requirement type, *Checking*. Here is a list of its attributes:

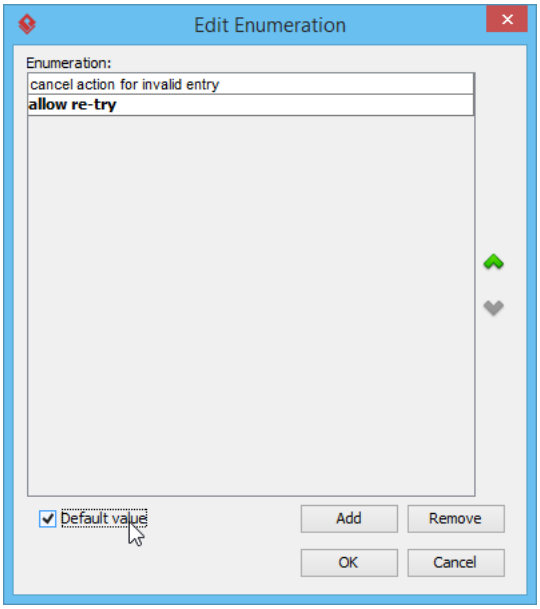
Attribute	Type	Description
description	Text	A description of the checking that needs to be performed.
policy	Enumeration	Determines the action to take when bad input is detected upon checking. You may allow the user to try again or just cancel the action immediately.
allowed-attempts	Integer	The number of attempts a user can make.

16. Add the enumeration values *cancel action for invalid entry* and *allow re-try* for the *policy* attribute. In most checks, you should allow the user to try again. Therefore, select "allow re-try" and check **Default value** at the bottom. This will set *allow re-try* as the default value, which will be automatically selected when you create a <<Checking>> requirement.

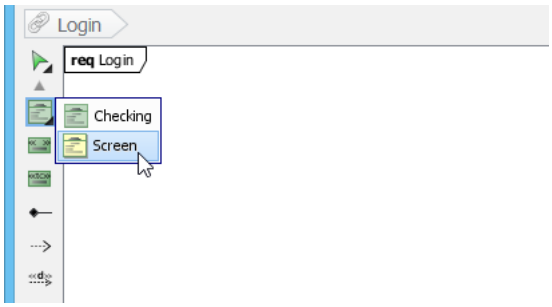


17. The requirement types are now configured. Click **OK** to return to the requirements configuration. Click **OK** again to confirm all the changes and return to the diagram.

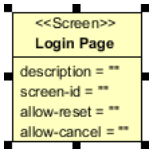
18. Now for the final attribute, *allowed attempts*. This is an attribute that requires a numeric input. Therefore, click **Add** and select **Integer Attribute**. Enter *allowed attempts* as the name.



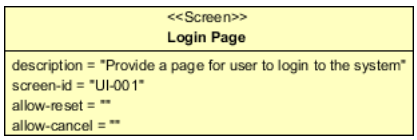
19. You can now create requirements with the new types. In the diagram toolbar, click on the **Requirement** tool and select *Screen*.



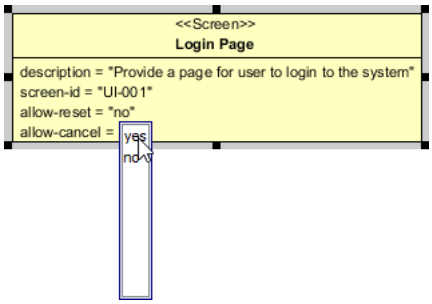
20. Click on the diagram to create a requirement. Name it *Login Page*. You will get a requirement like this:



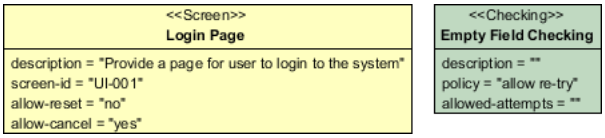
21. Double-click on the *description* attribute and enter *Provide a page for the user to log in to the system* as the description. Similarly, double-click on *screen-id* and enter *UI-001*.



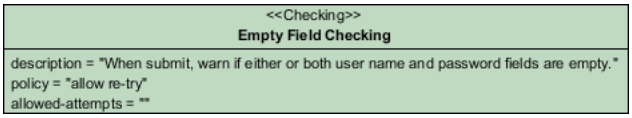
22. Remember that *allow-reset* and *allow-cancel* are enumeration attributes. If you double-click on *allow-reset*, you will see that the value is restricted to either "yes" or "no," as defined. Select *no* for *allow-reset* and *yes* for *allow-cancel*.



23. Now, use the *Checking* type to specify a requirement for validating form input. Click on the *Screen* requirement type in the diagram toolbar and select *Checking*. Click on the diagram to create a requirement. Name it *Form Field Checking*.



24. Double-click on the *description* attribute and enter *When submitted, warn the user if either or both the user name and password fields are empty*.



25. It is fine to let the user re-enter the correct email address. Therefore, just keep the *policy* as *allow re-try*. Leave *allowed-attempts* empty to indicate that there is no restriction on the number of attempts.

26. Create another <<Checking>> requirement named *Login Checking*. Set the description to *When submitted, verify and disallow invalid login.* and the policy to *cancel action for invalid entry*. Finally, you should have a diagram like this: