Kofax Kapow 10.3 Training and Certification

Module 10 – Advanced Repeat/Next Loops

Looping through pages with no Next button

Kofax Kapow™

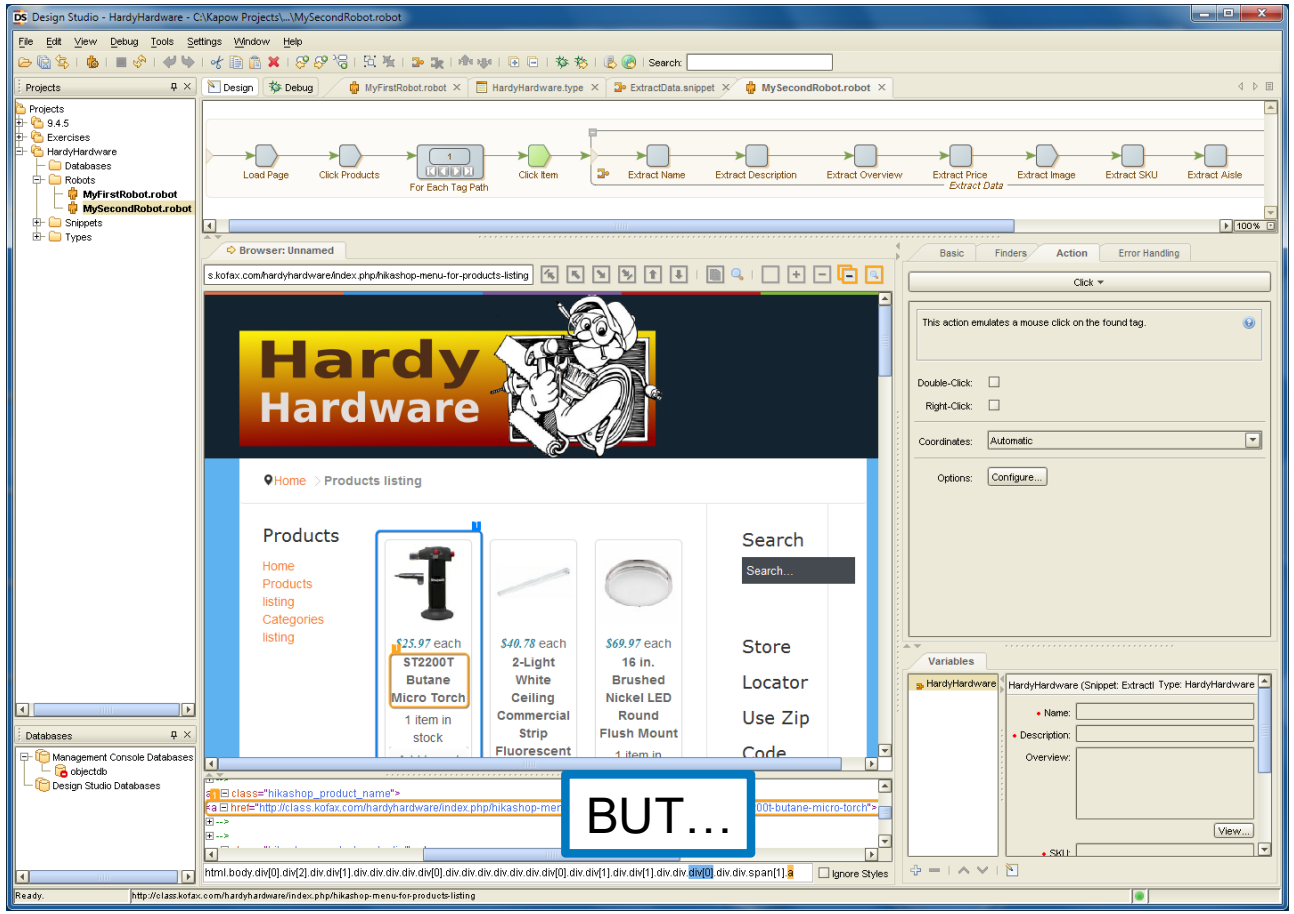
KOFAX

Module Overview

- More Advanced Repeat/Next Loops
- More Branching
- Simple Variables
- Assign Variable Step
- Extracting Attributes
- Extracting from a Table
- Named Tags

Extracting Data from Multiple Pages

- So far, we've set up a Robot that will loop through multiple items on multiple pages and return the data for those items.



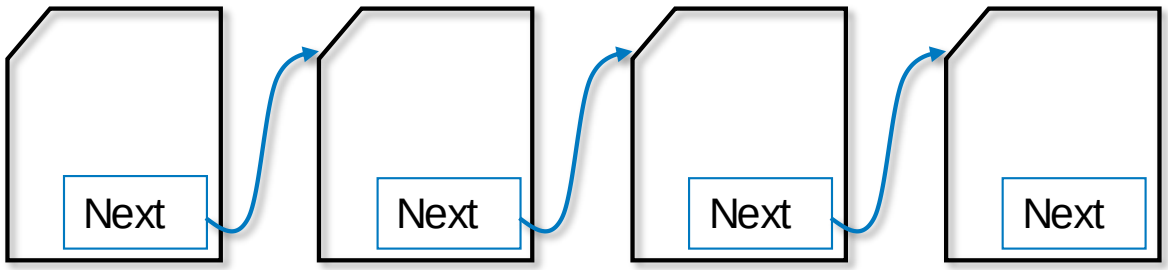
There are Multiple Pages

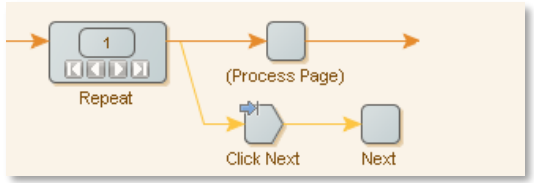
- If we go to the bottom of this page, you can see that is followed by page after page after page. Suppose we want ALL products listed. And notice, on this page, there is no convenient [Next] button!

- The answer is to have our robot loop through all pages. Once again, we will use a Repeat/Next Loop.

A Quick Review - Repeat Loops

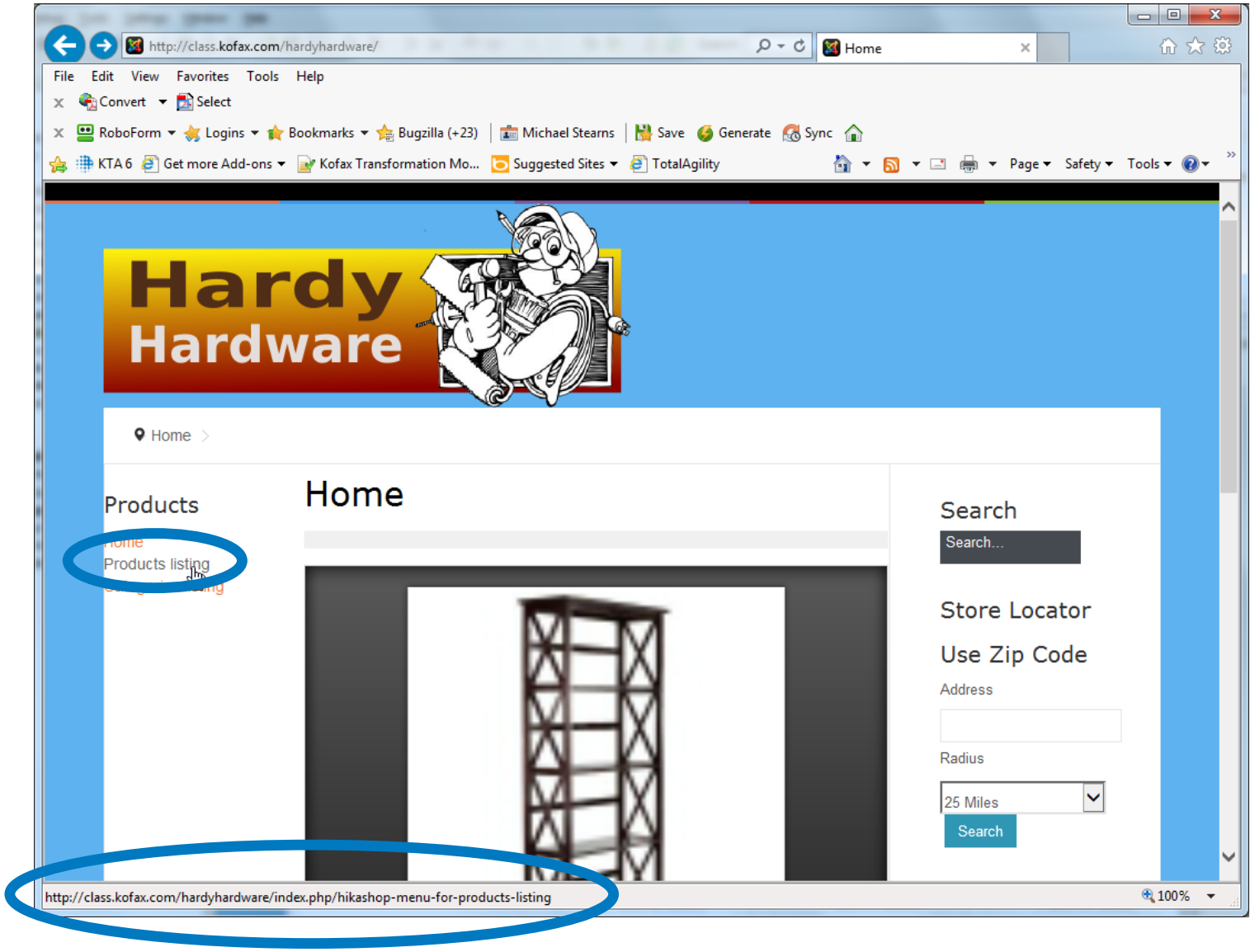
- A Review: Repeat/Next Loops...
  - This action creates a repeat loop together with the Next action.
  - The Repeat action marks the start of the repeat loop. In a subsequent step, another iteration of the loop can be requested using a Next action.
  - The windows, pages, etc. at the Next step will be sent back to the Repeat step and will become the output from the Repeat step in the next iteration. If no Next step is executed in a given iteration, that iteration will be the last one, and the repeat loop will end. But first…let's build our Robot.



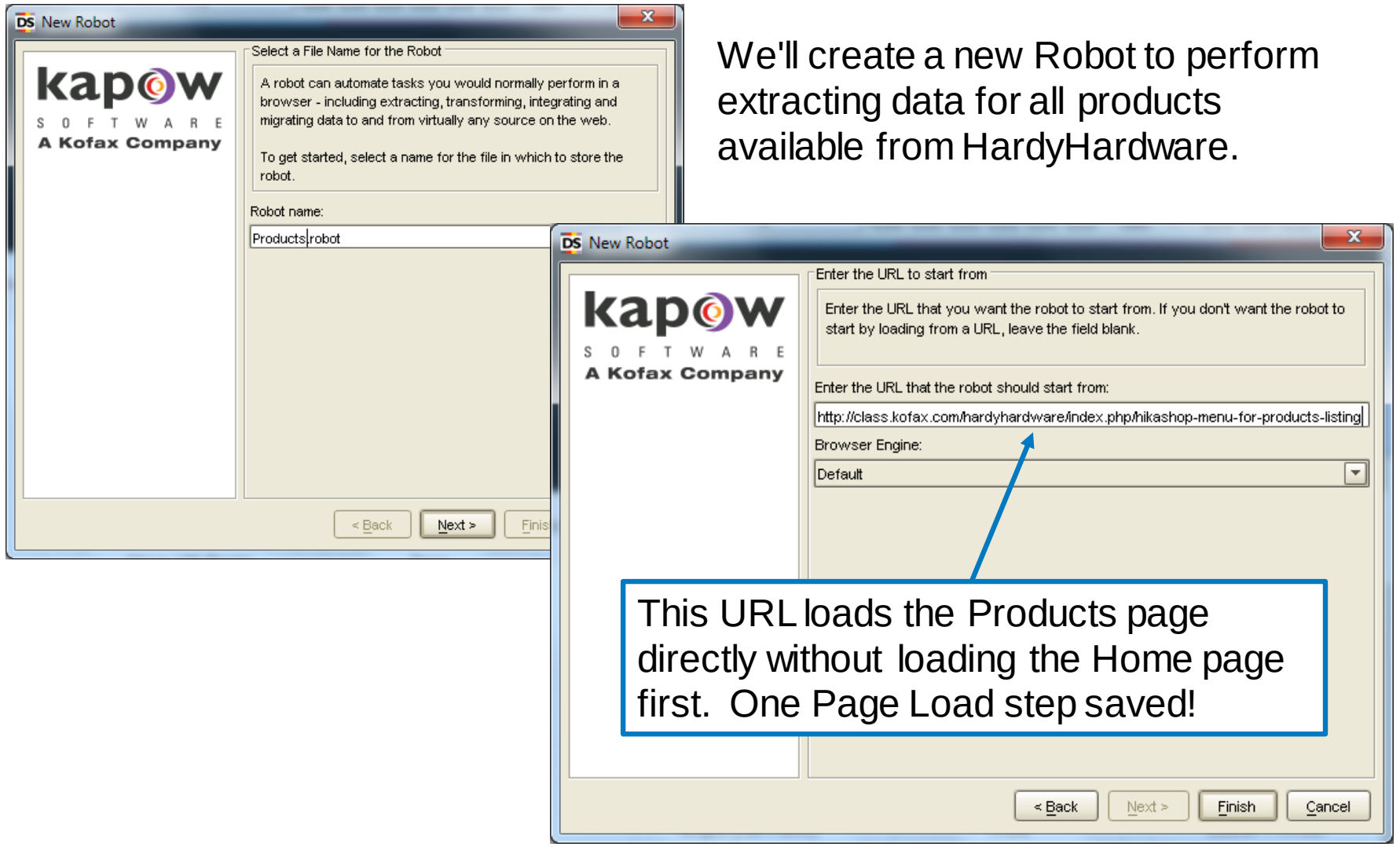


We Can Load the Products Page Directly

- Instead of loading the home page and then clicking on Products, we can make this Robot less "expensive" in terms of KCUs by loading the Products page directly. The URL is shown on the status bar of the browser.



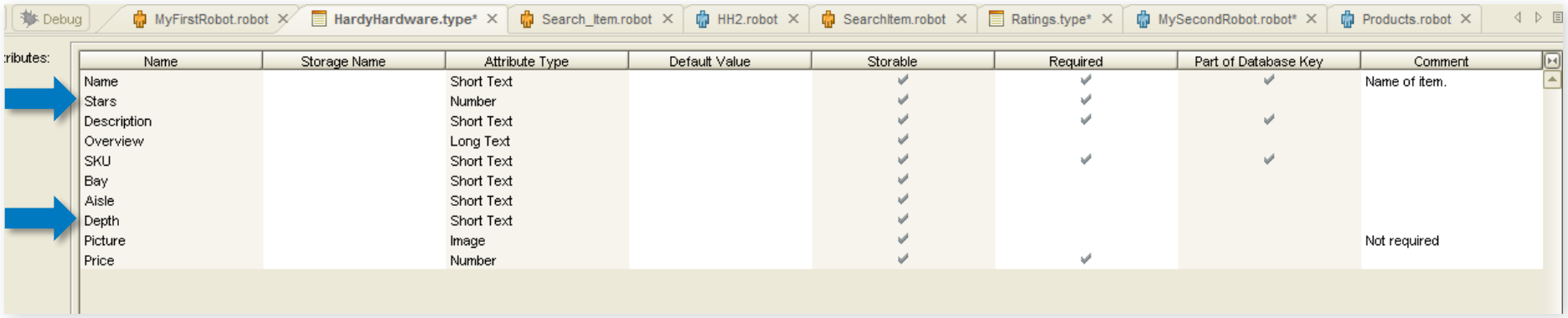
Create a New Robot



We'll create a new Robot to perform extracting data for all products available from HardyHardware.

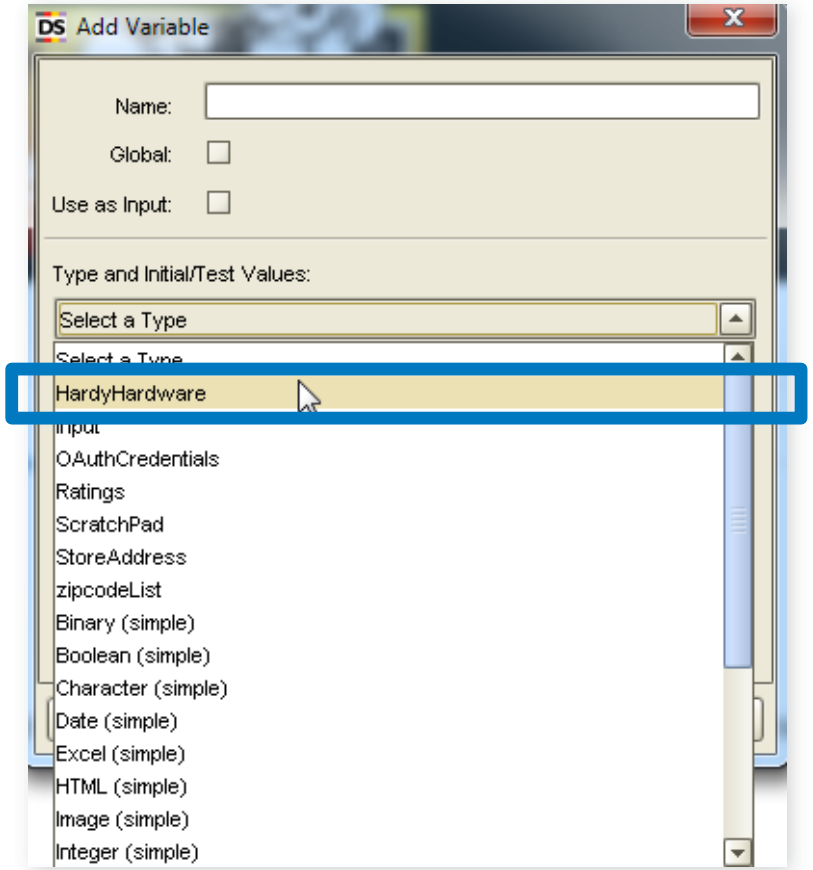
Add Attributes to Type

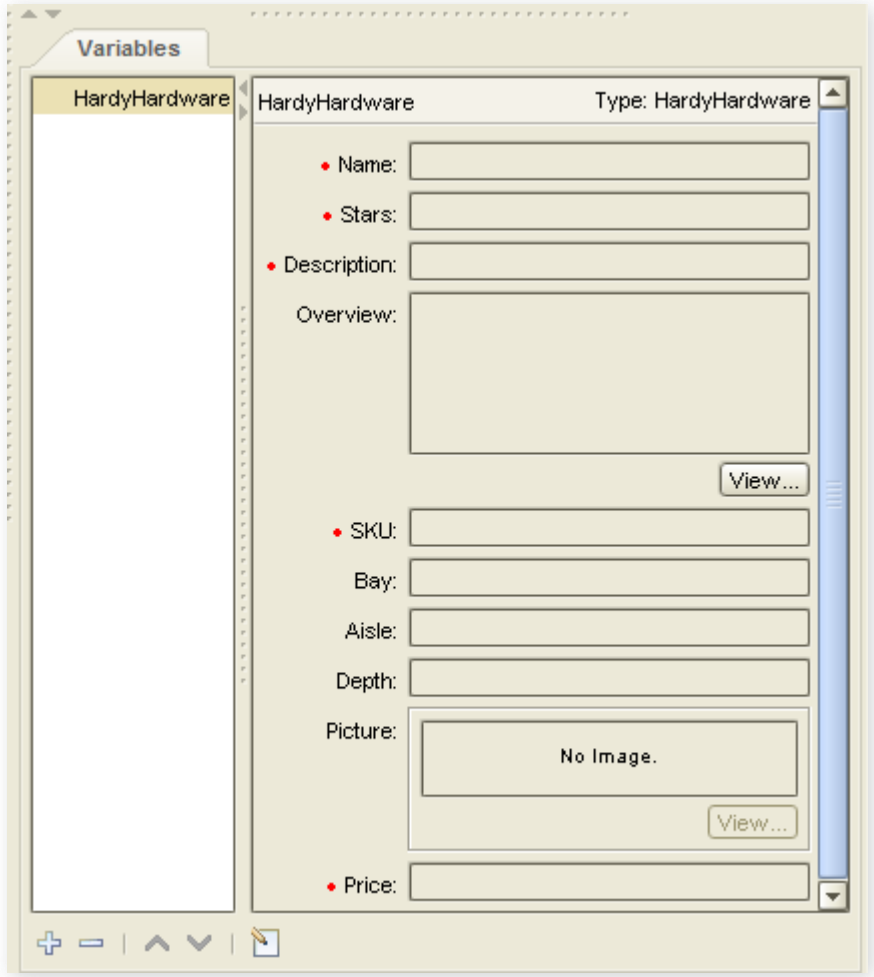
- Let's imagine we also want to extract the rating (number of stars) ★★★★★ for each product.
- We'll also extract the depth measurement of each product (we'll do this because it's a little more challenging, and after all, this is a training class).
- First, we have to add two new attributes to our HardyHardware.Type



Remember how to add Attributes?

Add Variable

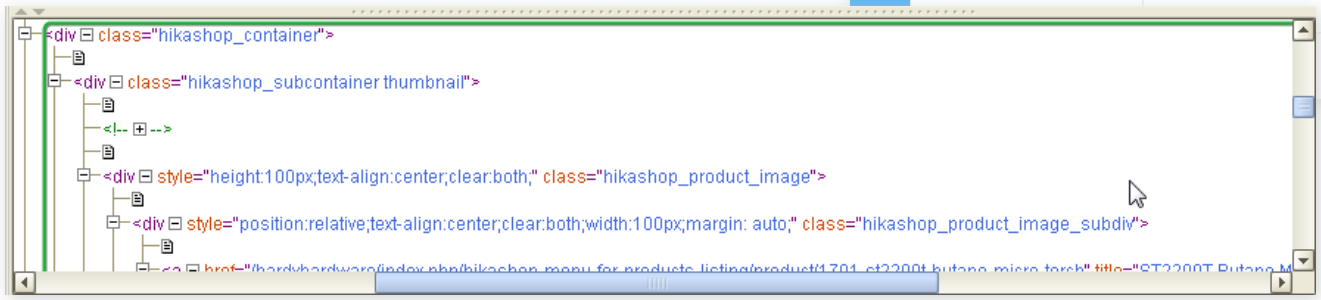




Now we'll go back to our new Robot and add a Variable to contain the data that will be extracted.

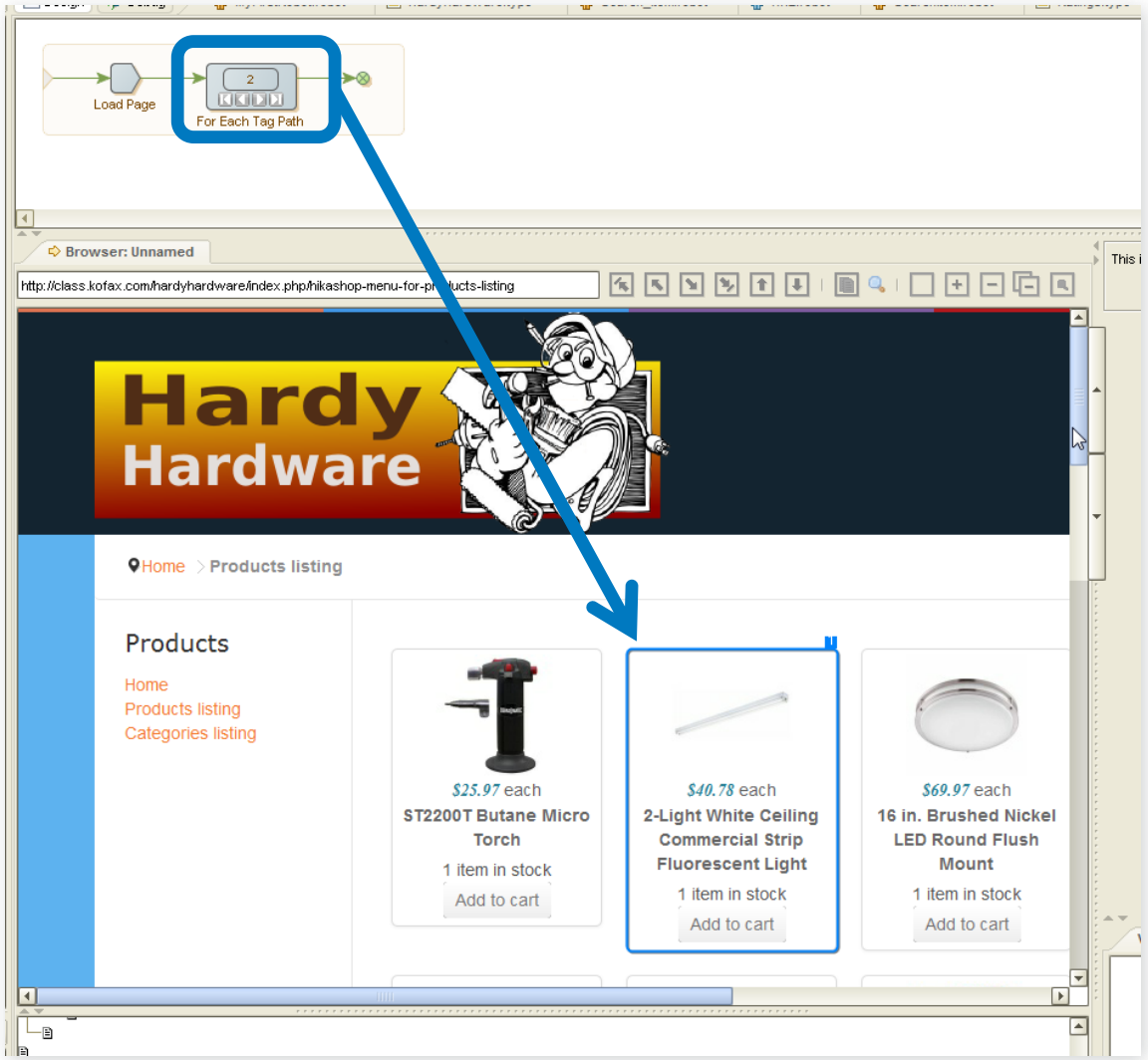
Loop Through Each Item on a Page

- We'll want to loop through each item displayed on the Products page, so we've clicked on the first item on the page.
- Then we expanded our green box up one level and selected Loop for Tags With Class "hikashop_product_image."

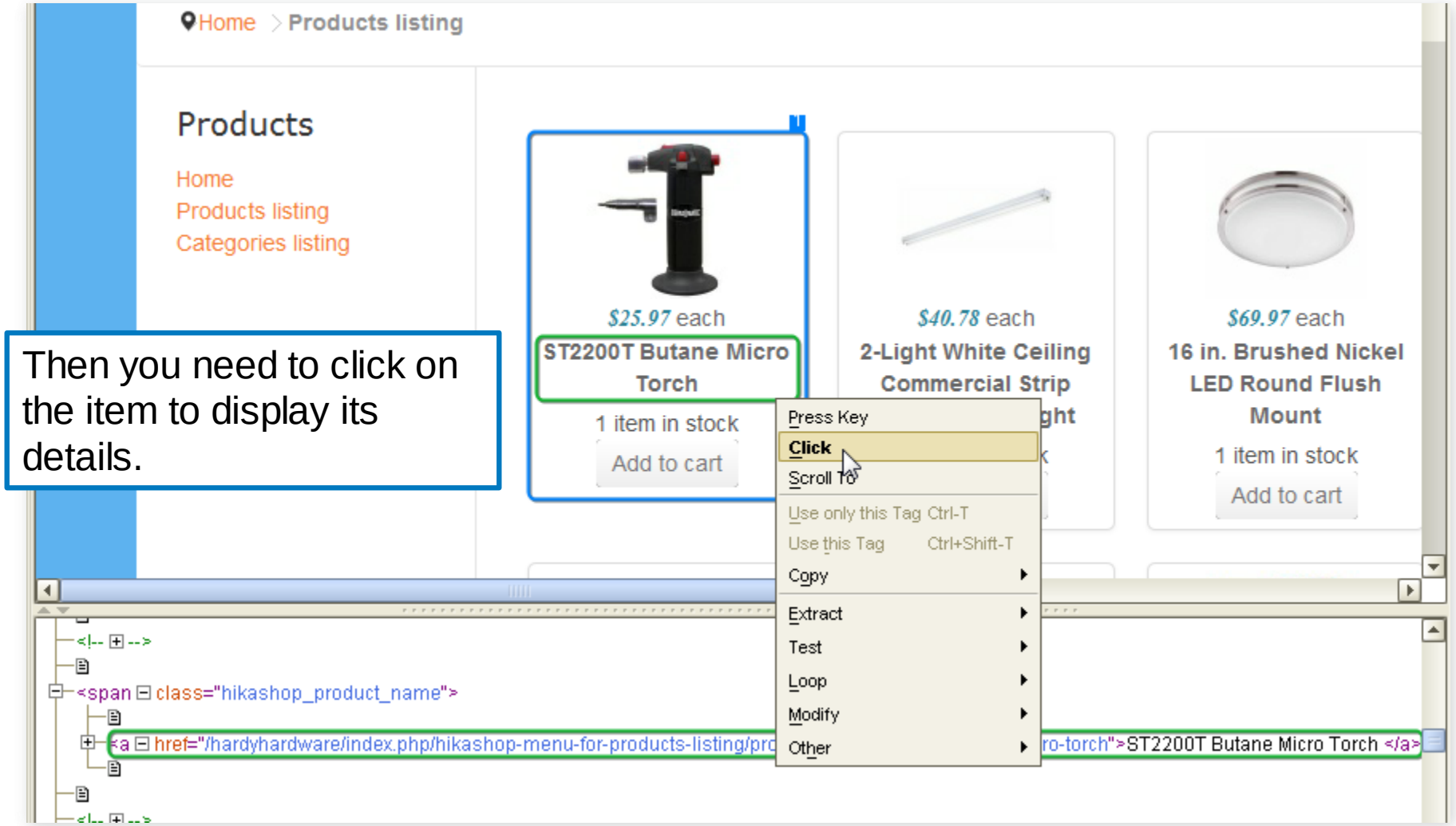


Test Loop

- With the end step selected, you can use the forward arrow in your loop step to test that the loop works.
- Here, we've selected the second iteration of the loop.



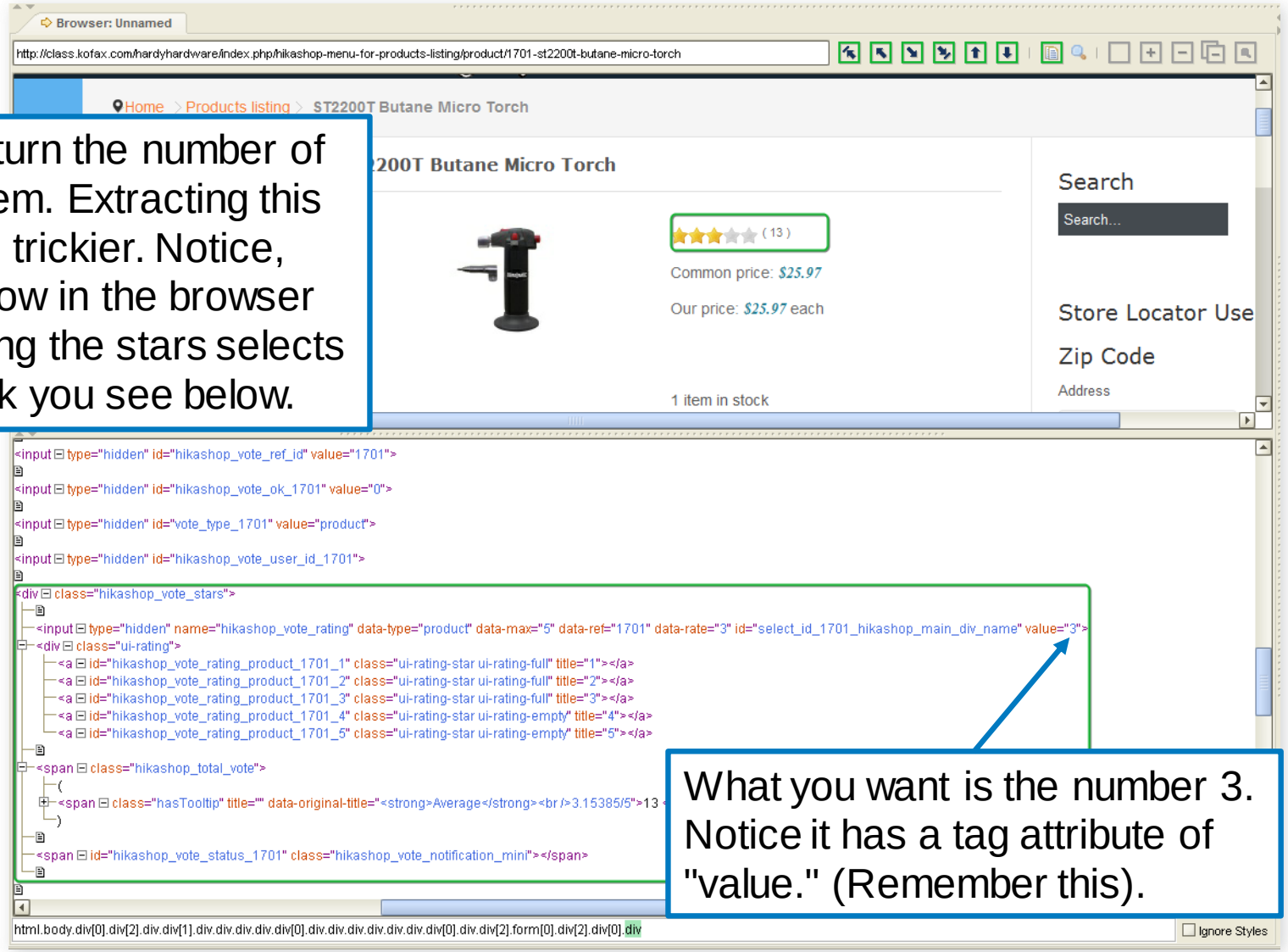
Add Click Step to Load Item Details



Then you need to click on the item to display its details.

Extract Rating (Stars)

We want to return the number of stars for the item. Extracting this rating is a little trickier. Notice, selecting the row in the browser panel containing the stars selects the entire block you see below.

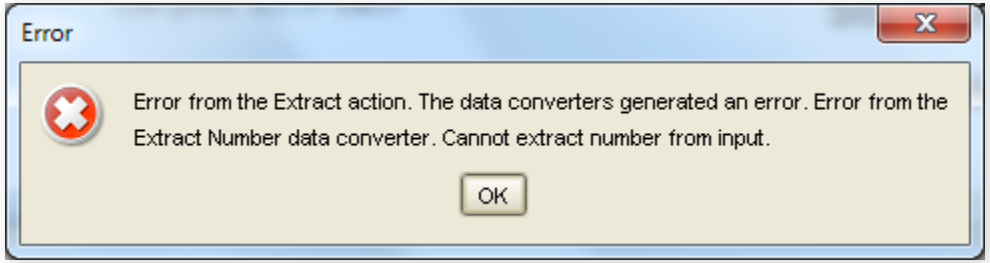


What you want is the number 3. Notice it has a tag attribute of "value." (Remember this).

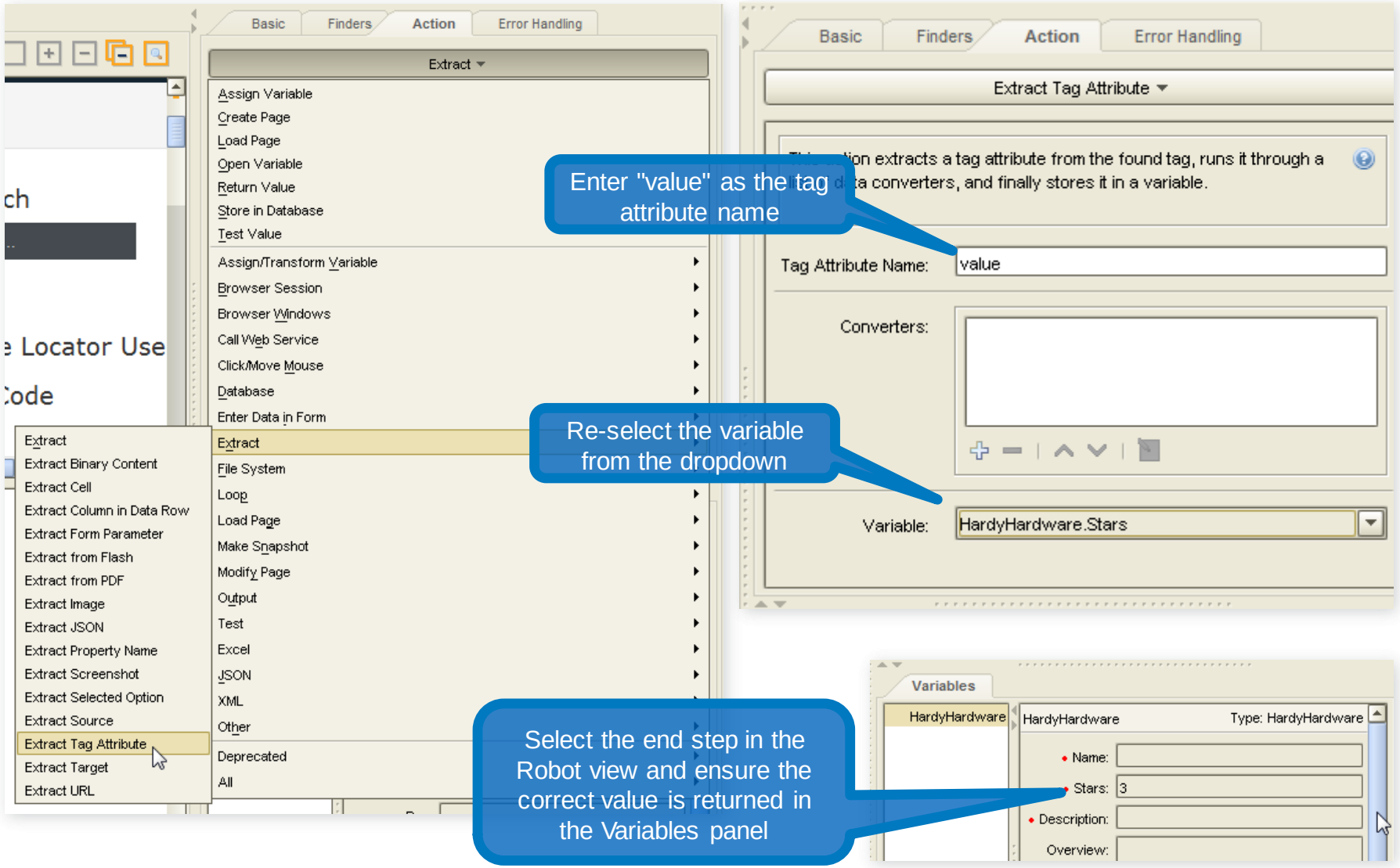
Select From the Source Code Instead

- Select the input tag shown here (which will return all the content on that line).

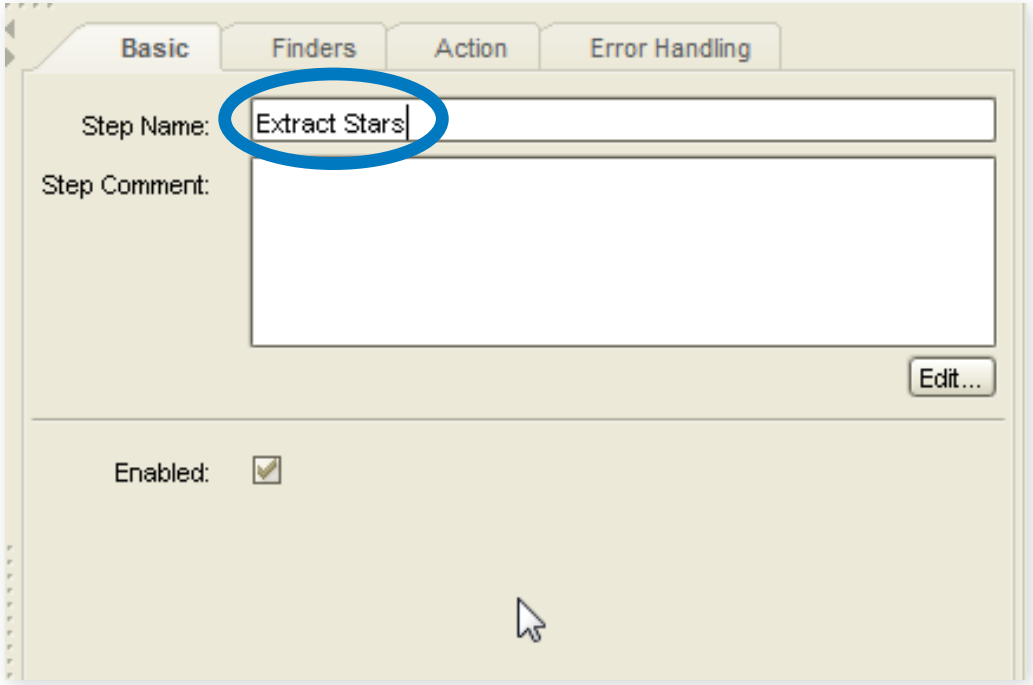
The Data Converter used to extract a number will generate an error. We'll fix that in the next step.



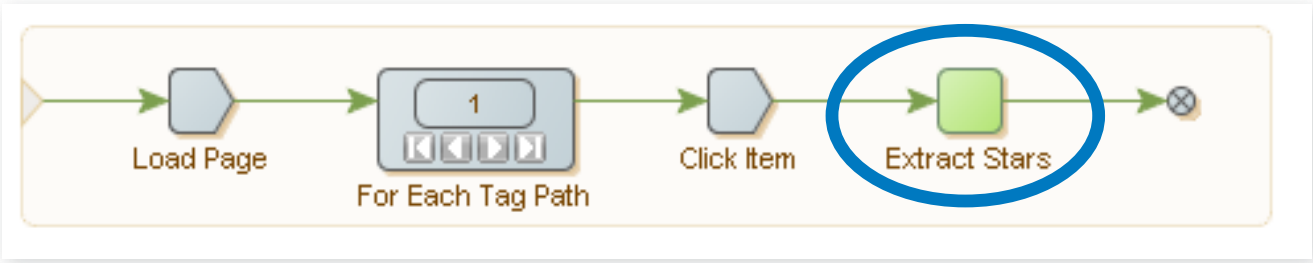
Change Extract Step to Extract Tag Attribute



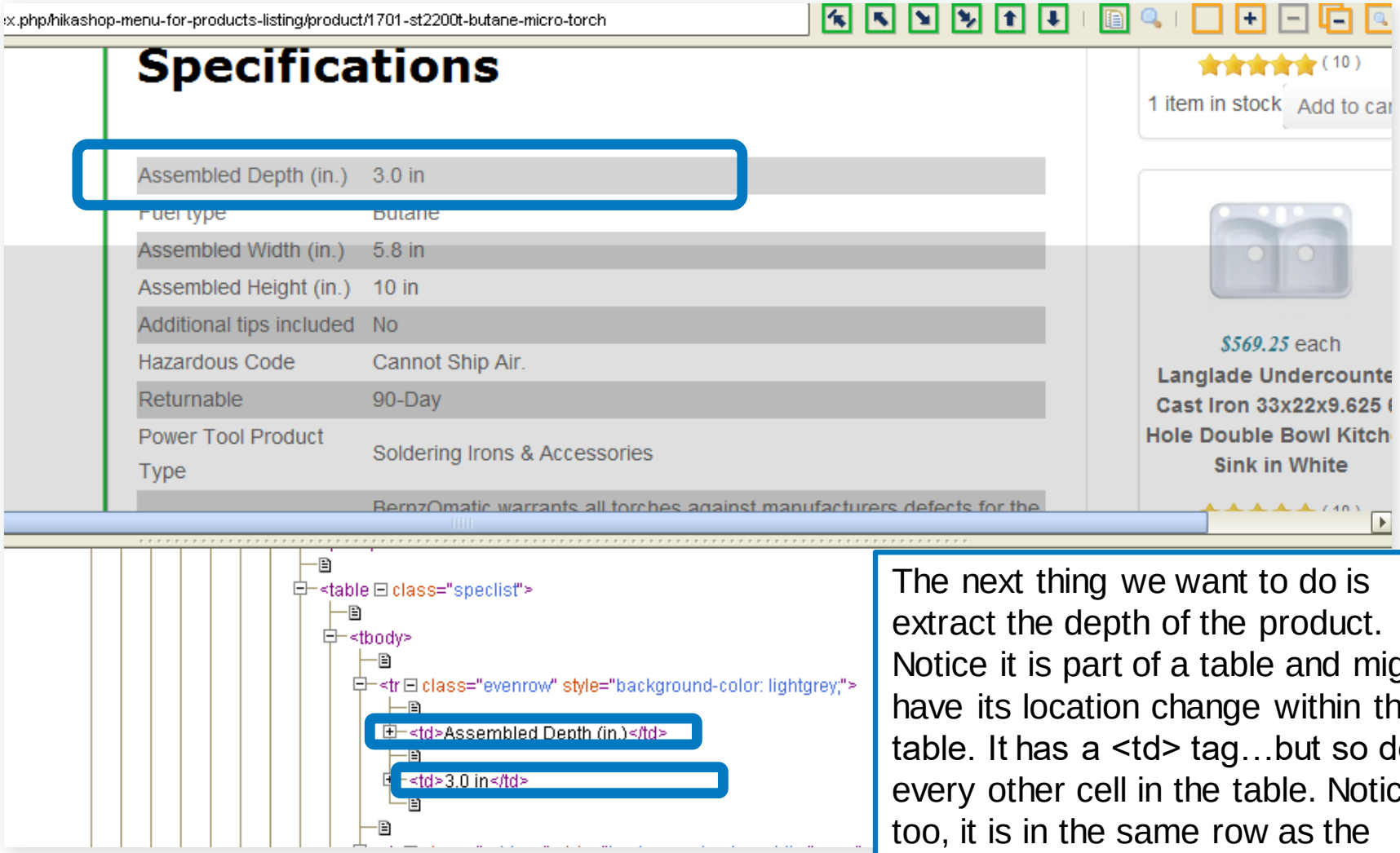
And Change the Name of the Step to Make More Sense



Remember, the name of the step in your Robot is available on the Basic tab of that step's properties.

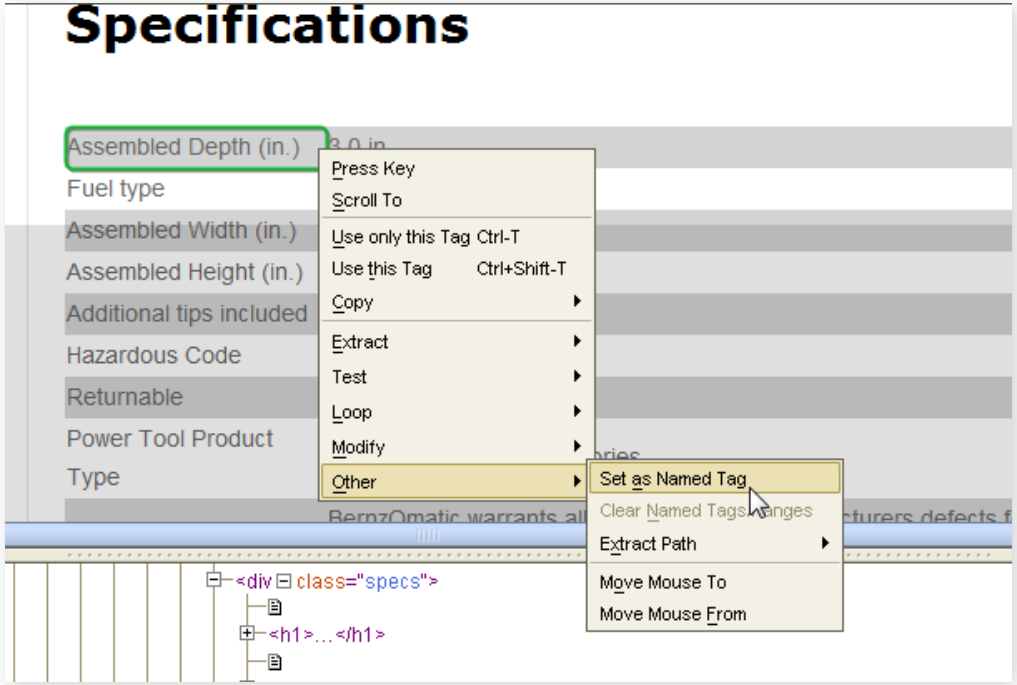


Extract Depth Measurement



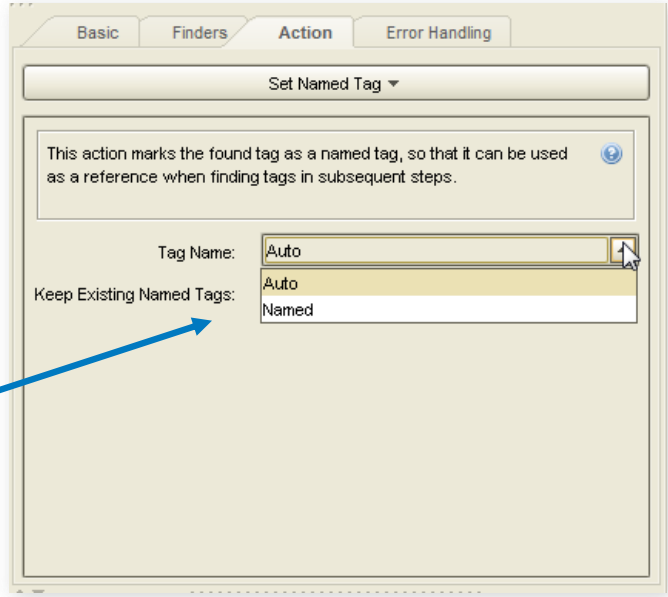
The next thing we want to do is extract the depth of the product. Notice it is part of a table and might have its location change within the table. It has a <td> tag…but so does every other cell in the table. Notice too, it is in the same row as the label, "Assembled Depth (in.)"

We Can Find the Depth We Want Using a Named Tag

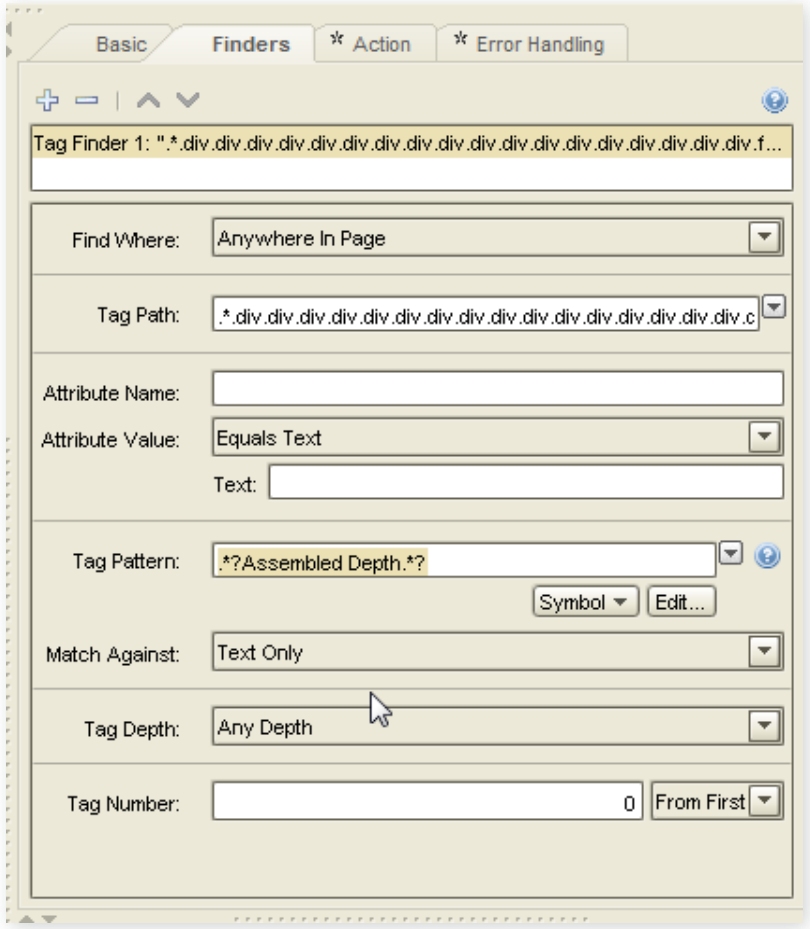


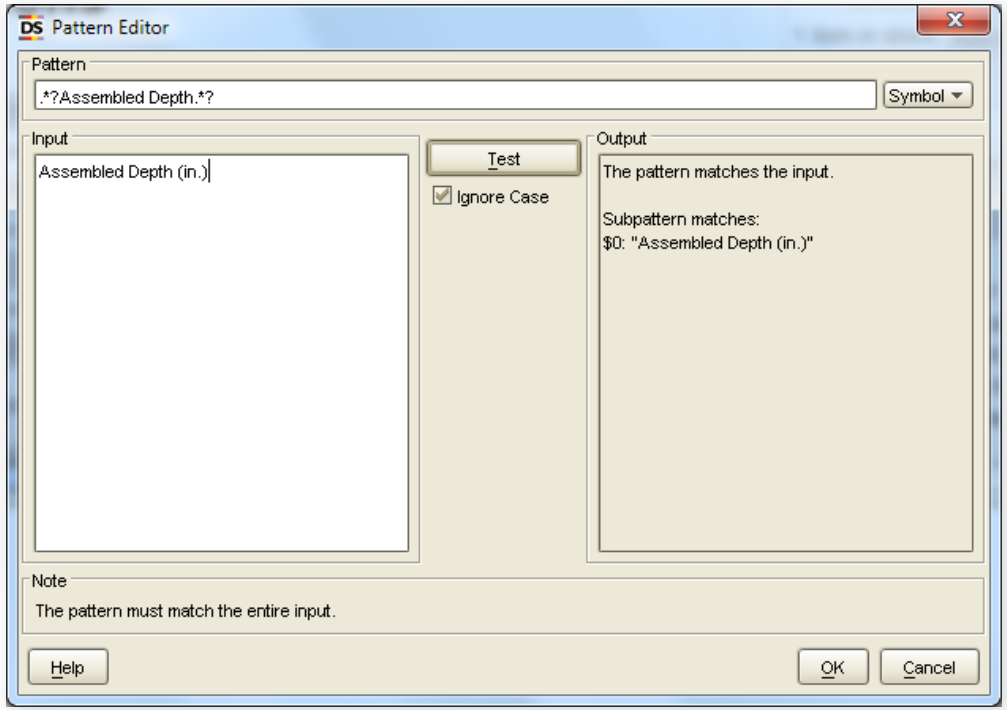
Named Tags can be used to find other tags.

Auto gives the tag a name (which is number). The first Auto-numbered tag will have number 1, the next number 2 etc. **Named** gives the tag a fixed and explicitly stated name and can make it more obvious what it is finding. We'll name ours "Assembled Depth." We'll give the step the same name as well.

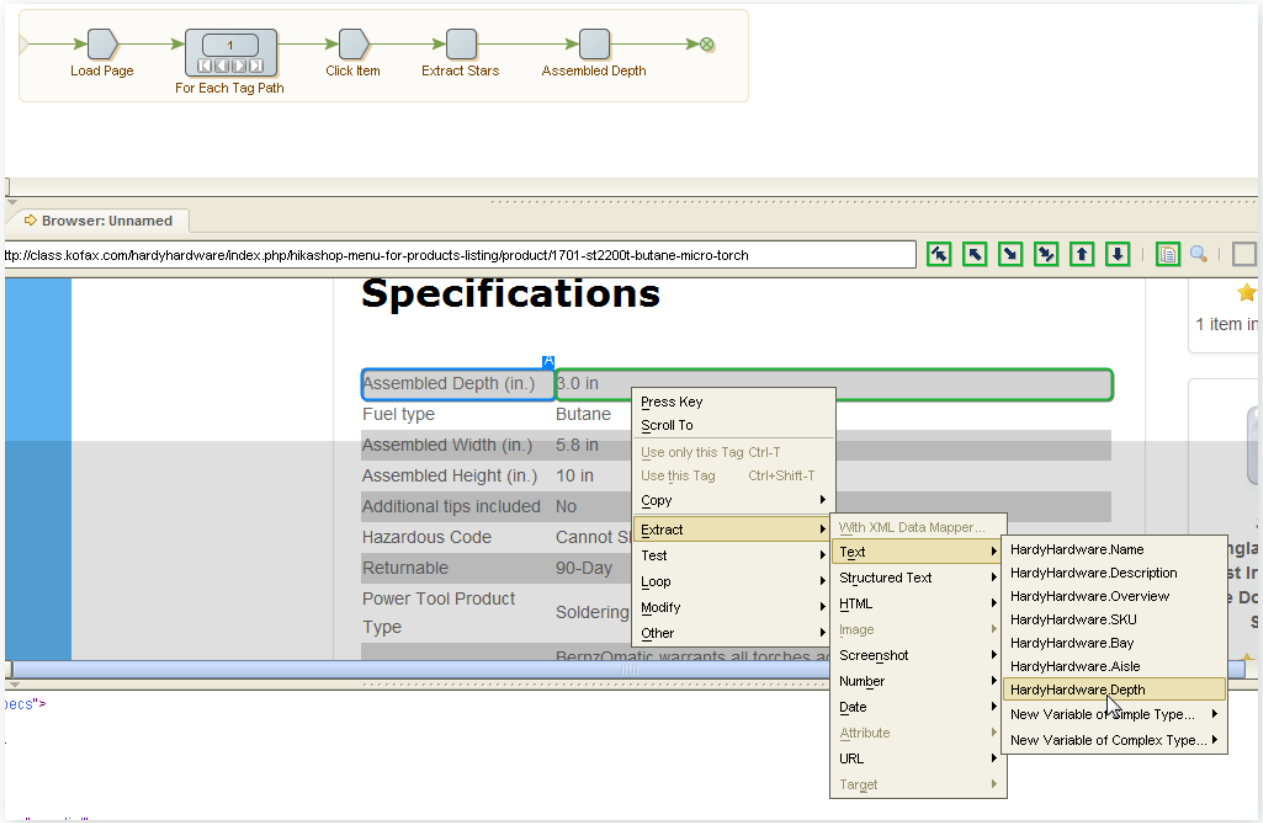


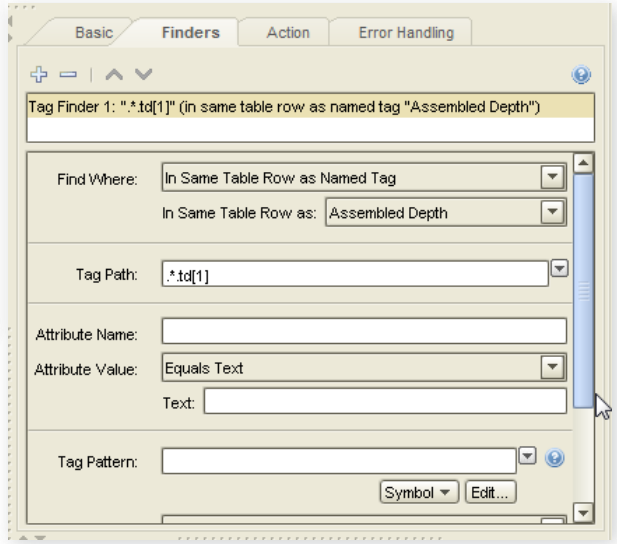
Specify a Tag Pattern to Find





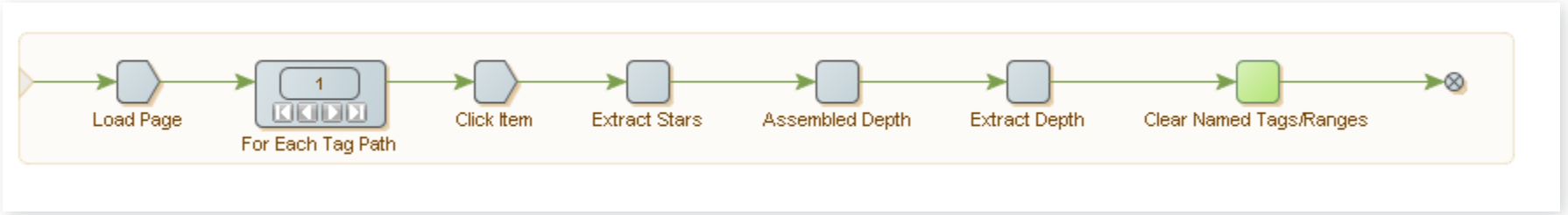
Extract Depth



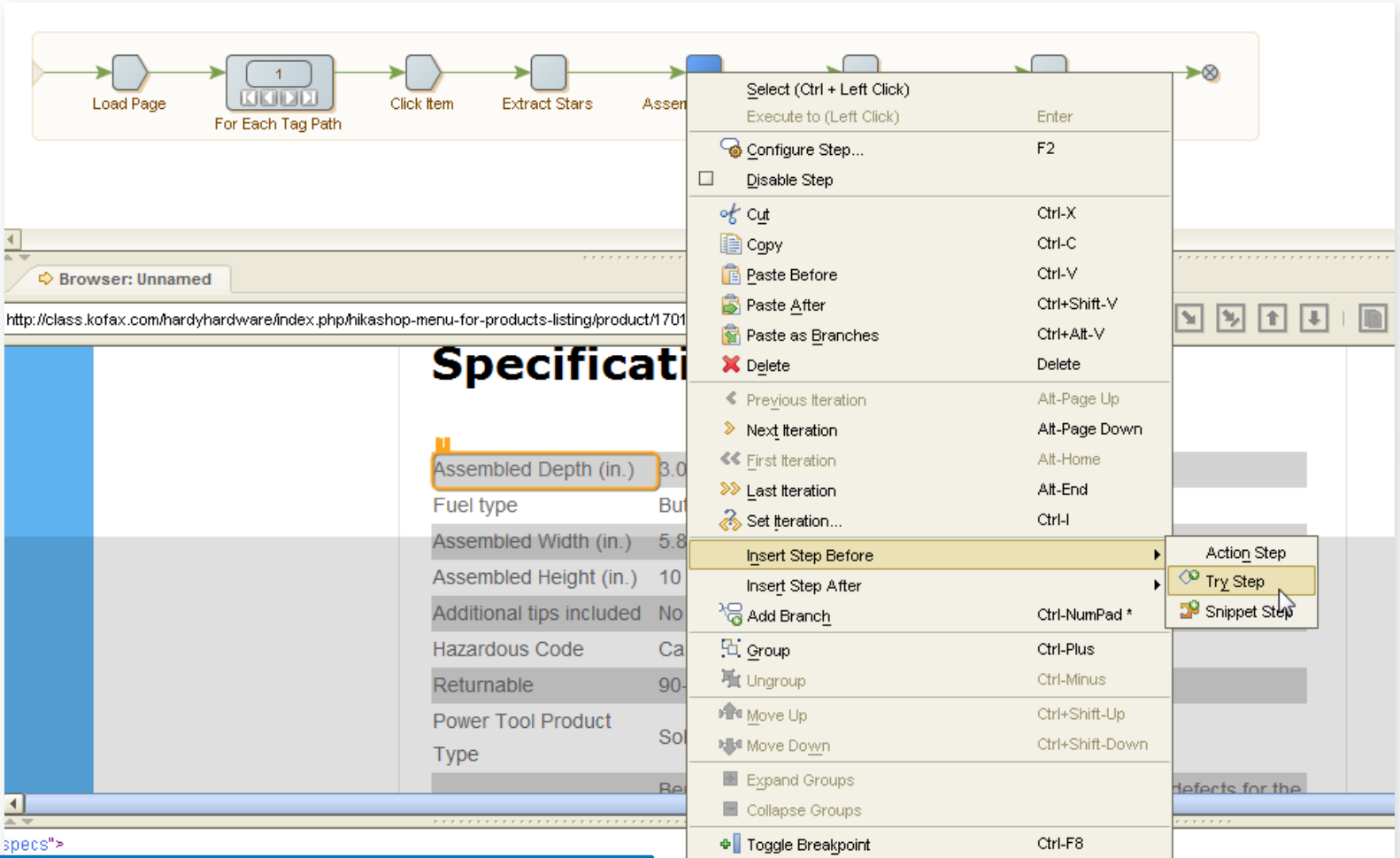


Now We Need to Clear the Named Tag

- This action unmarks a selected named tag or range, or all named tags and ranges, so that these will no longer be named in the subsequent steps.

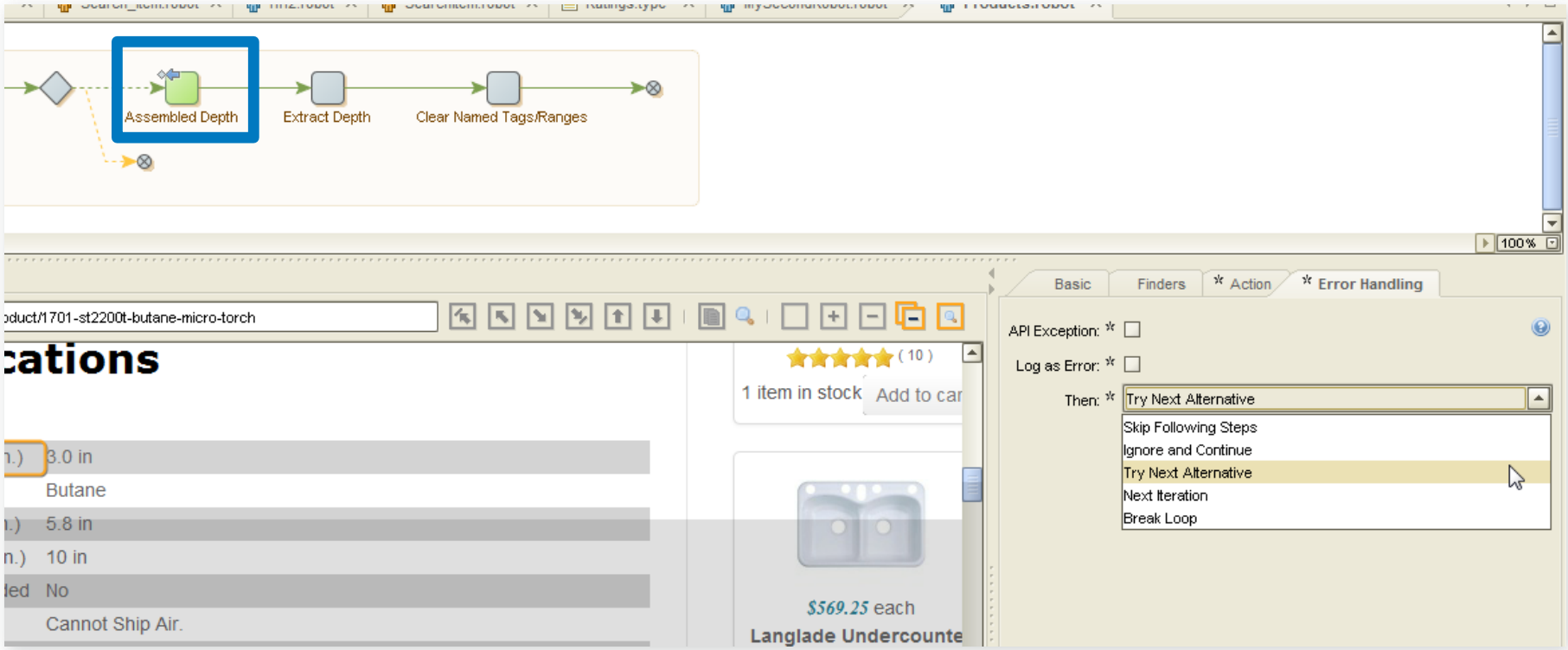


But What if "Assembled Depth" Doesn't Exist



Let's add a Try Step before the "Assembled Depth" find named tag step.

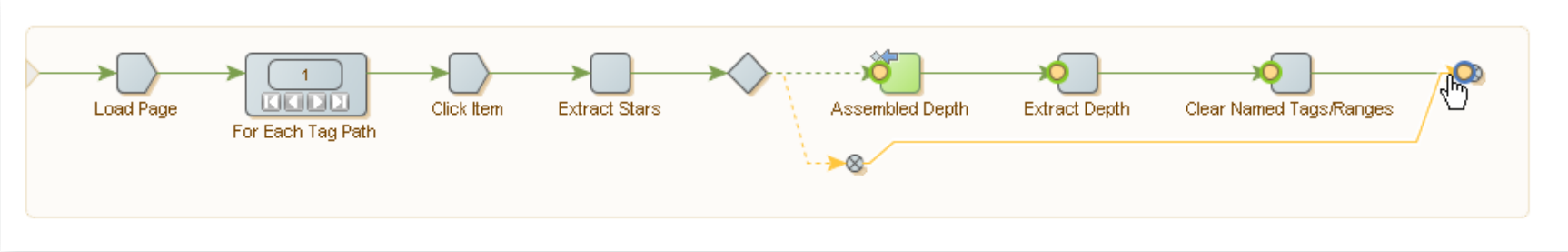
And Change "Assembled Depth" Error Handling



So if the Assembled Depth tag finder finds nothing, the step will go to the next alternative branch and end. But we want to do some other extraction after trying to extract depth…so….

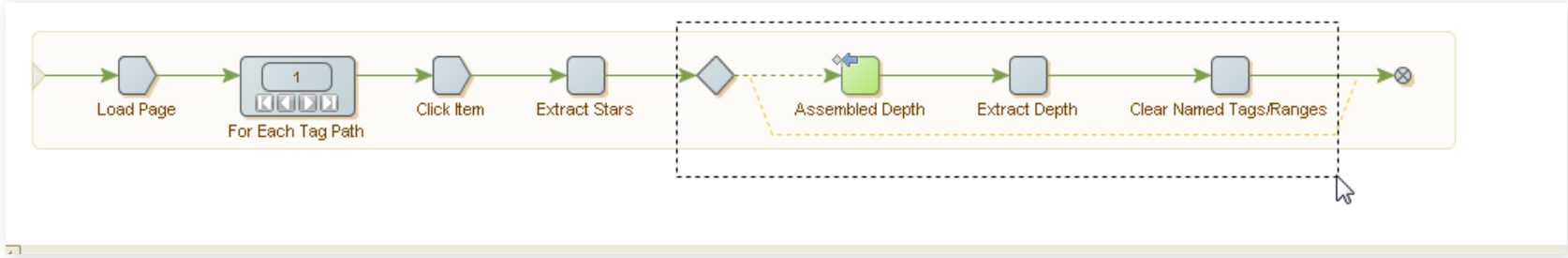
Join Try Step Branch to Skip Extraction of Depth

- Simply drag the end step of the branch to the end step above as shown. If no "depth" is found, the extraction steps for depth are skipped, and the Robot goes directly to the main end step. But we'll set up our other extraction in front of it as show on the next slide.

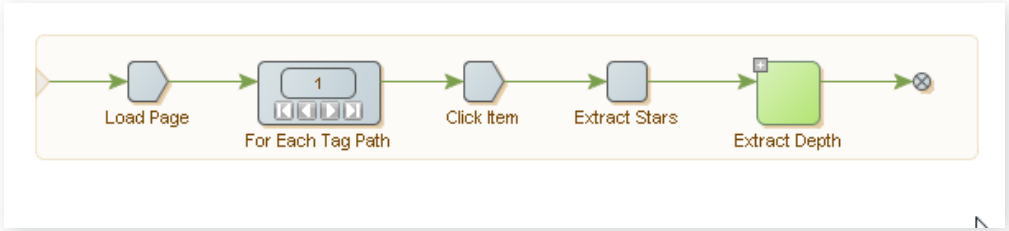


Create Group to Simplify Robot

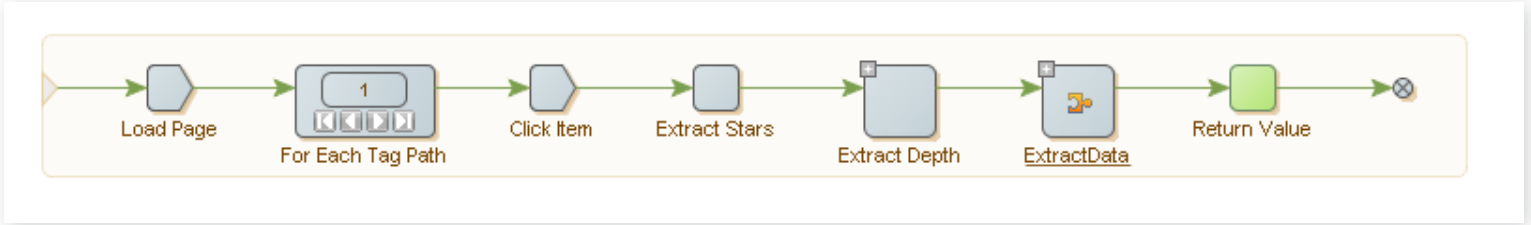
- Before we add the other extraction, let's create a Group for the steps we just created to extract the assembled depth of the item.



- Collapsed, this makes our Robot simpler to view.

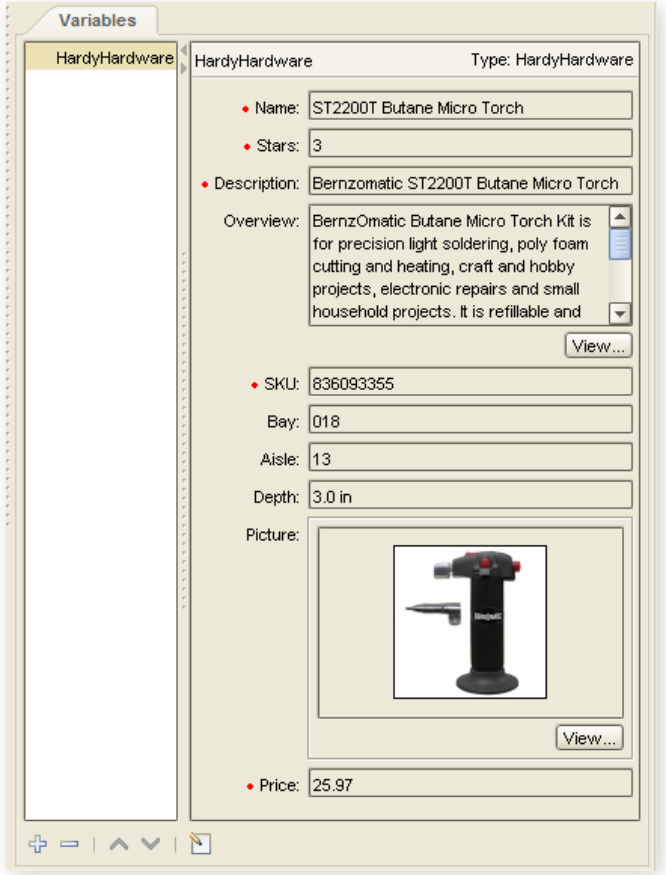


Let's Add Our Extract Data Snippet to Get the Other Data

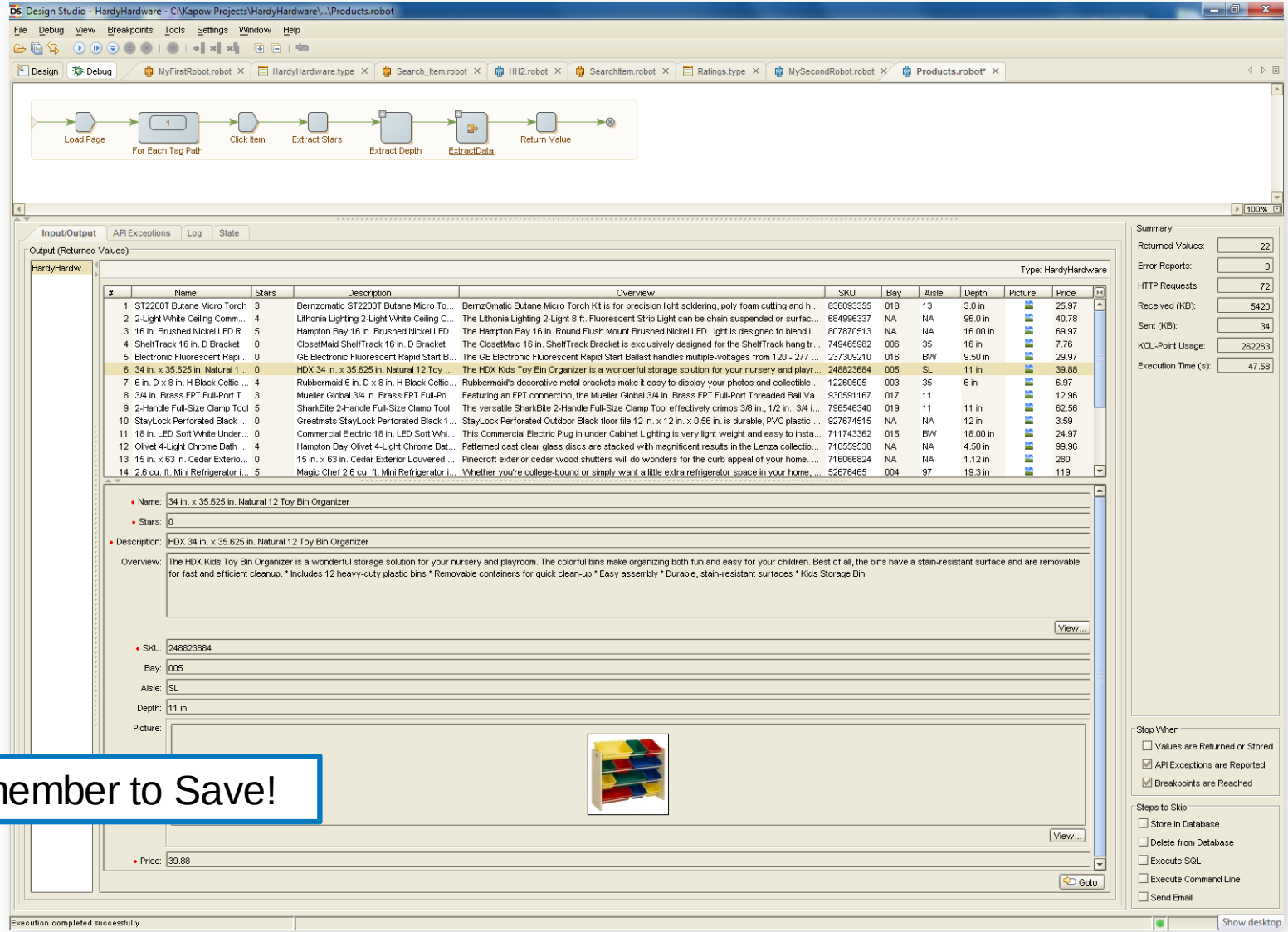


We'll insert the Snippet we created in another Robot along with a Return Value Action Step (for testing) as shown above.

Notice that all the data we wanted to extract is shown for the first item on our first page in the Variables panel.



And Test in Debug Mode (One Page Only…for Now)



Remember to Save!

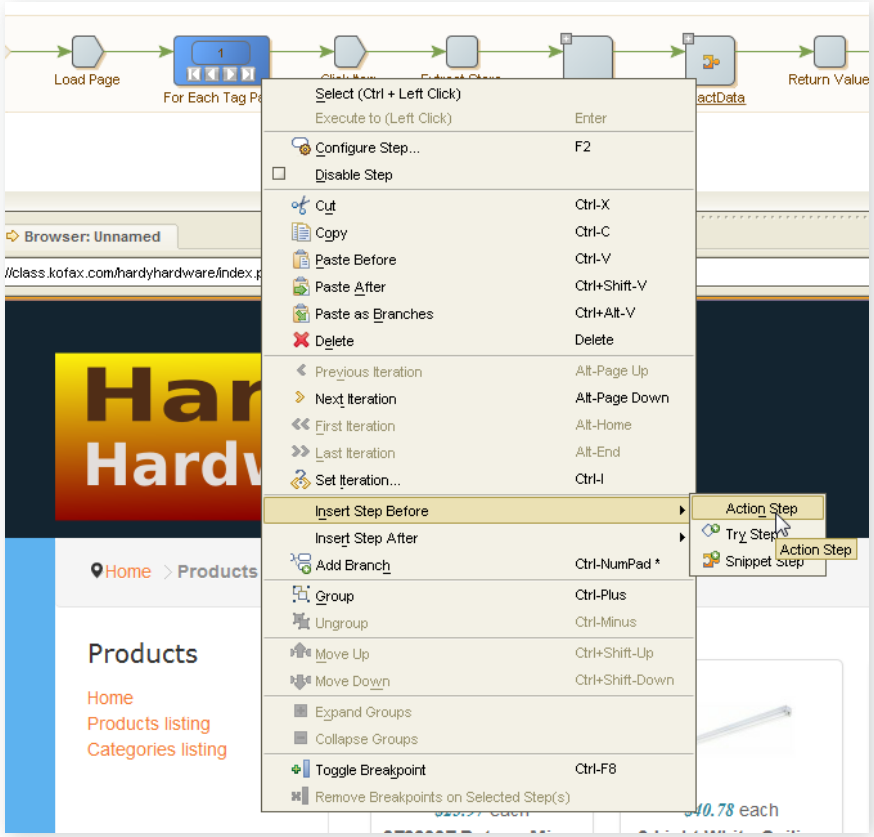
Repeat/Next Loops – A Little More Advanced This Time

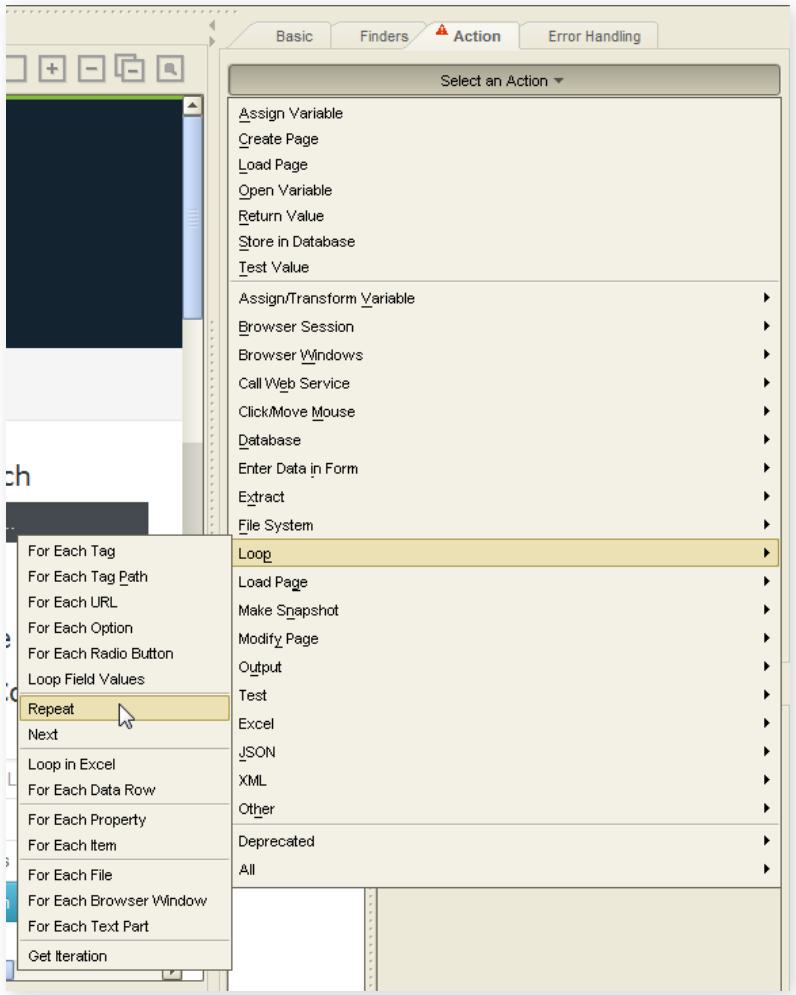
Now we want to go back and insert a Repeat/Next Loop to iterate through the 68 pages of products on this web site. You already know how to do that if there is a [Next] button. But none exists on the "Products" pages.

Here's how we're going to do it. After adding our Repeat Loop step…

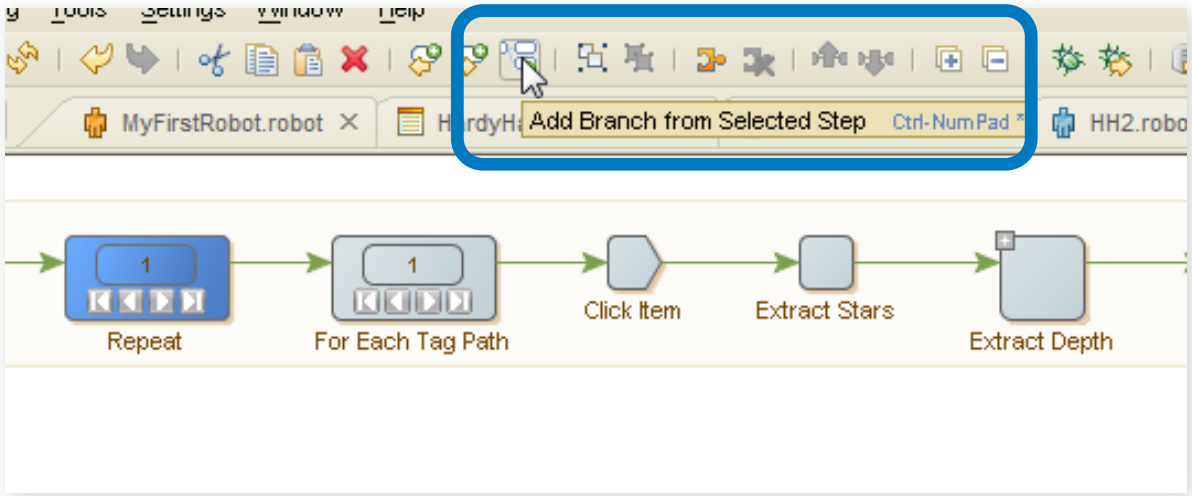
1. Create a new simple Variable that will contain the page number. It will only be used for internal processing and will never be output. Remember, Simple Variables can never be used for input or output.
2. Create a new "Assign Variable" step that will increment the page number by one for every iteration of our Repeat/Next loop step.
3. Create a "Click" step to click on the number of the page. Set error handling to break the loop if nothing is left to click on.
4. Create a Next step to go back to the Repeat step.

Add "Repeat" Step Before "For Each Tag Path" Step

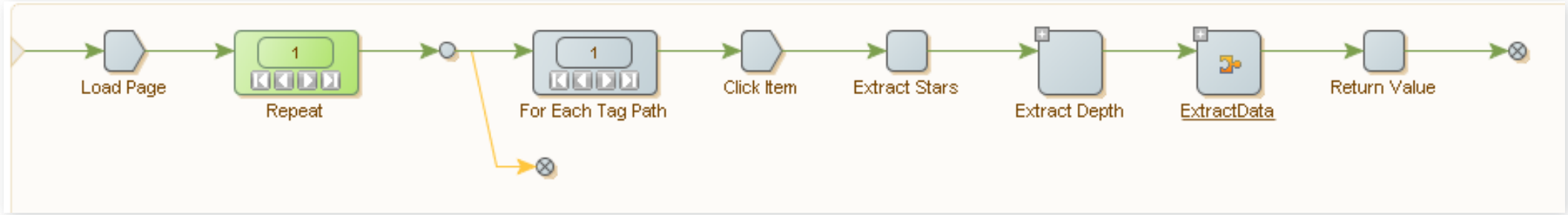




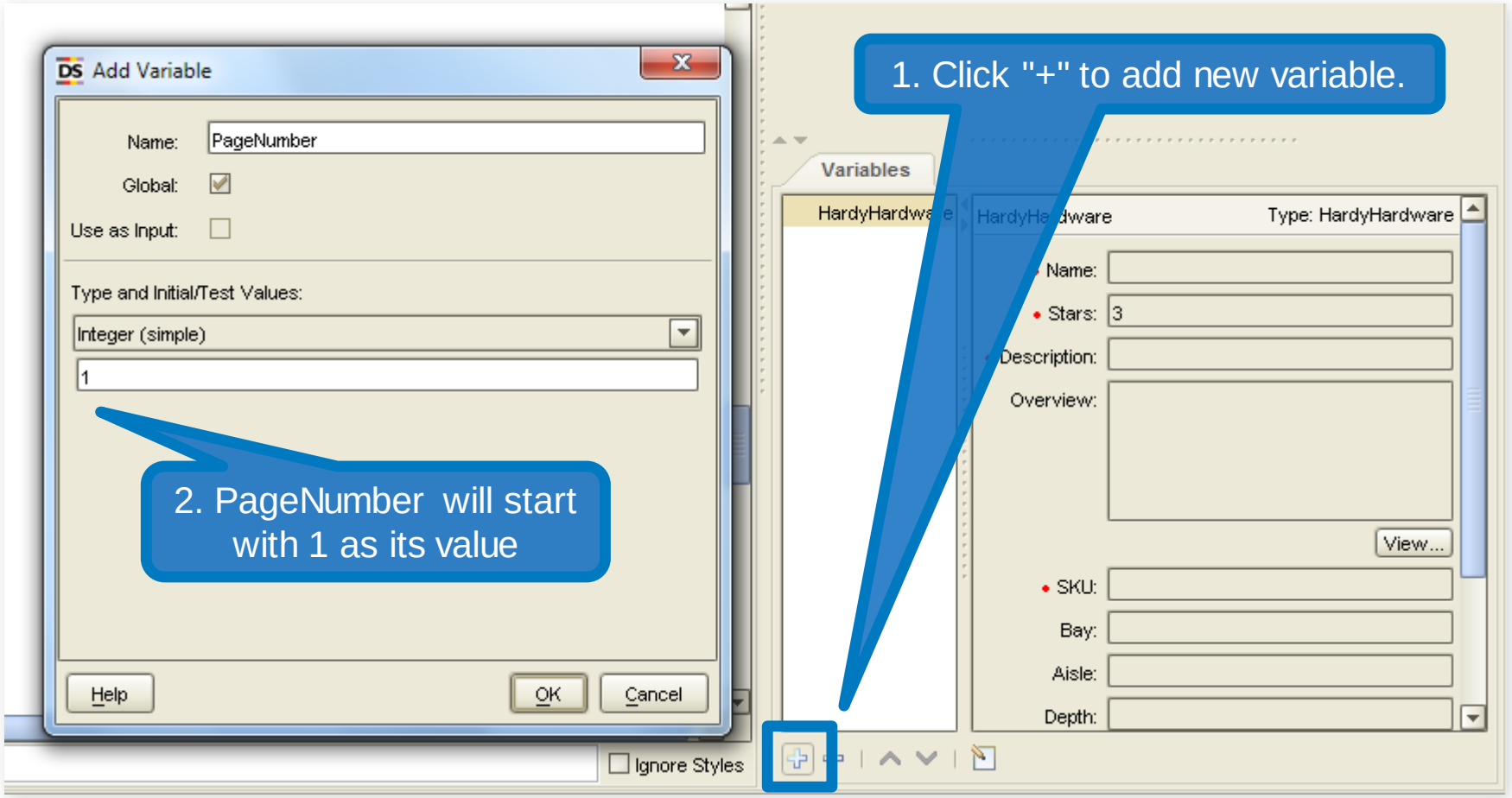
Add Branch from Current Step



Branch has been created. This is where we will add the balance of our steps.

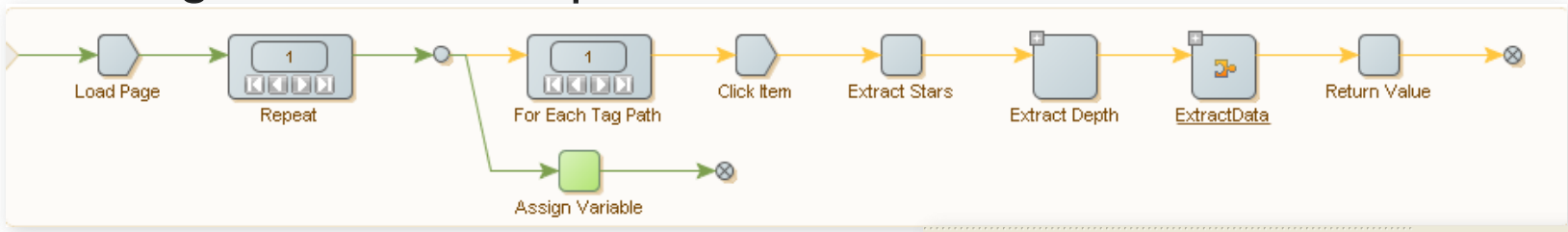


Add a New Variable



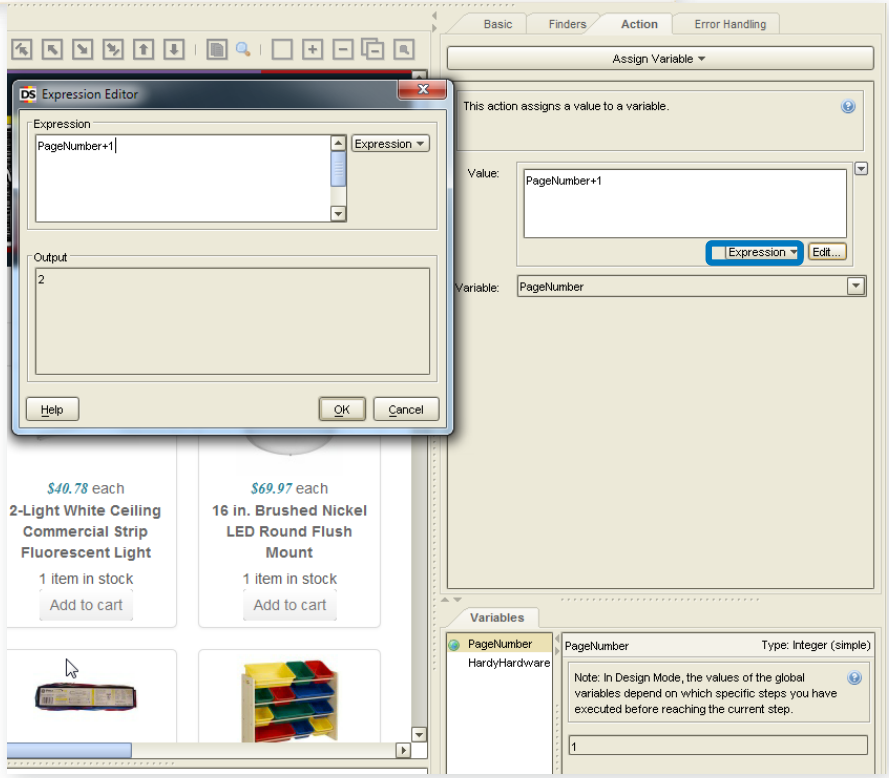
Add Assign Variable Step

- Add a new "Assign Variable" step on the branch.



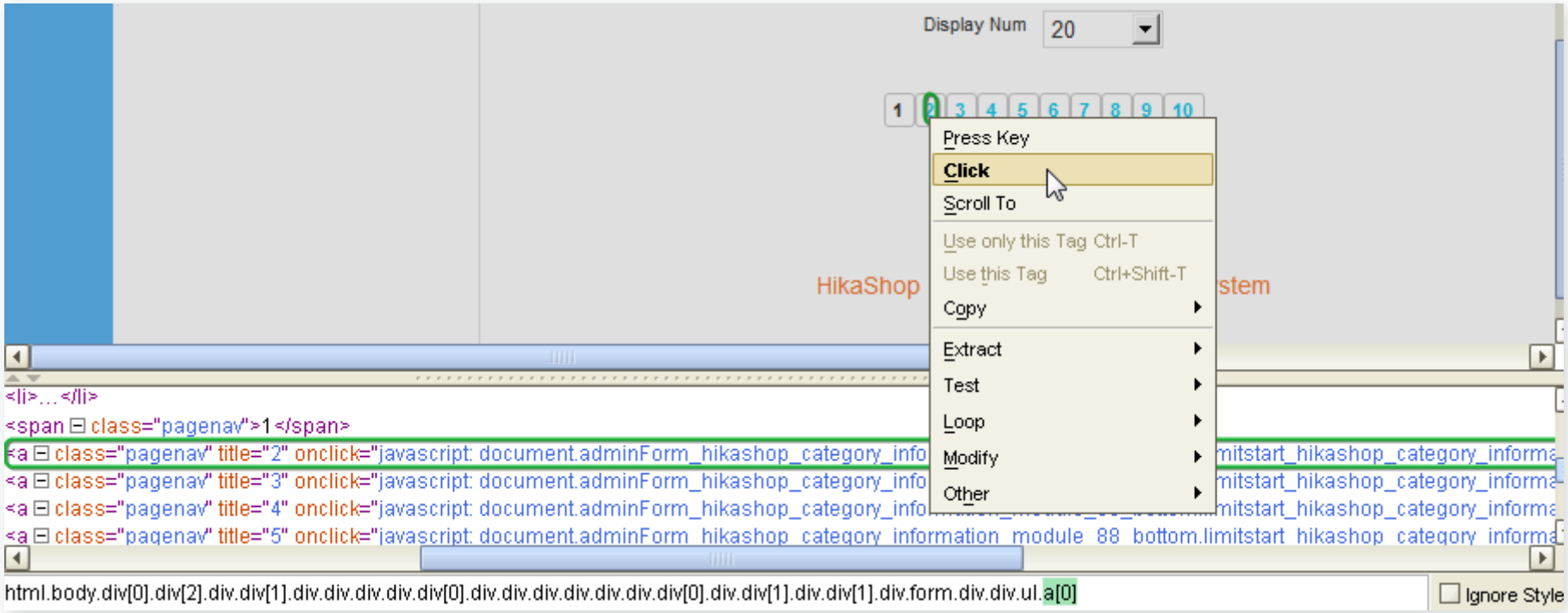
Set the Action properties for the step to use an expression that will take the value stored in the PageNumber variable and increment it by 1.

You do that by creating a simple expression as shown here.



Add Click Step

- Then we will want to add a "Click" step to click on the number.



Configure Finders for Click Step

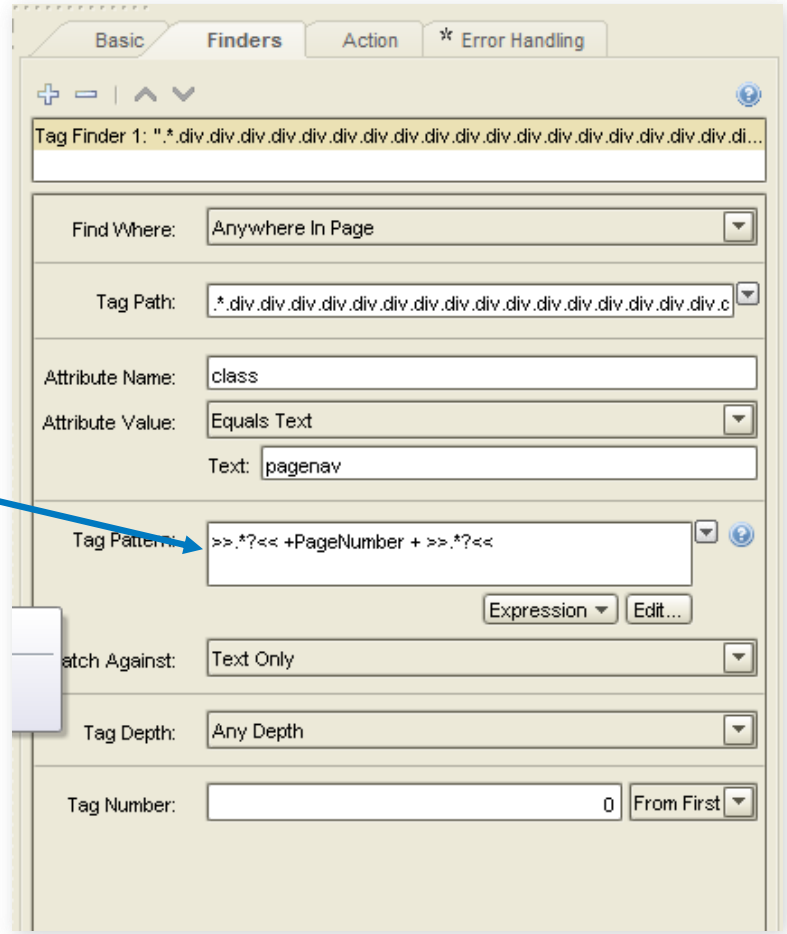
```
<span class="pagenav">1</span>
<a class="pagenav" title="2" onclick="javascript: document.adminForm_hikashop_category_information_module_88_botto
<a class="pagenav" title="3" onclick="javascript: document.adminForm_hikashop_category_information_module_88_botto
```

We need to include an expression that includes the updated PageNumber as a Tag Pattern.

Because expressions must contain the *entire* pattern represented, we can represent it this way:

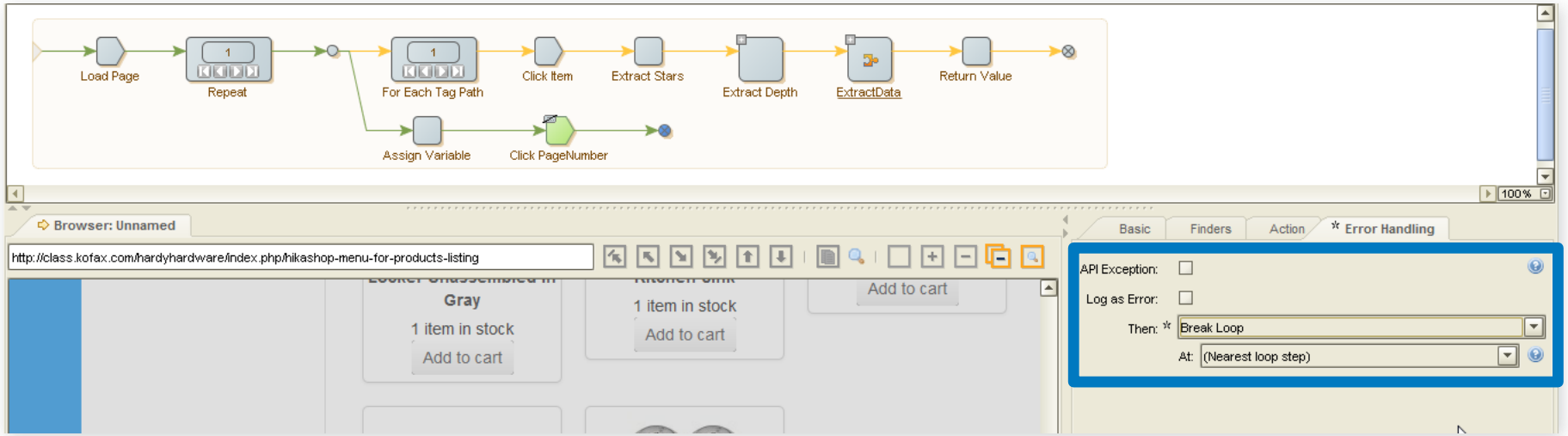
>>.*?<< + PageNumber + >>.*?<<

>> << is a constant containing anything. Here, we used **.*?** meaning any combination of characters or numbers, zero or one time.
That is concatenated with the value of our **PageNumber** variable, which is then concatenated by everything that follows until the next tag.



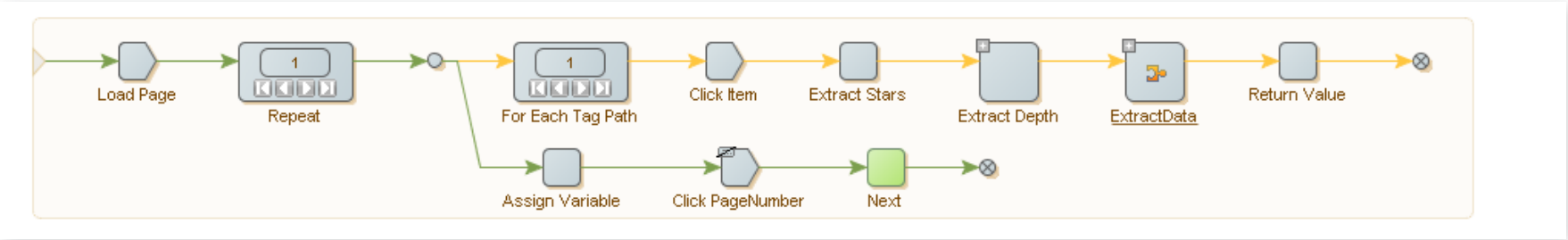
Set Error Handling for Click Step

- And we want to set the click step to break the loop if no number matching the value of "PageNumber+1" is found…like after the last page has been returned by our Robot. We do not want our Robot to return an error if this occurs, so we've unchecked "API Exception" and "Log as Error."



Add Next Step

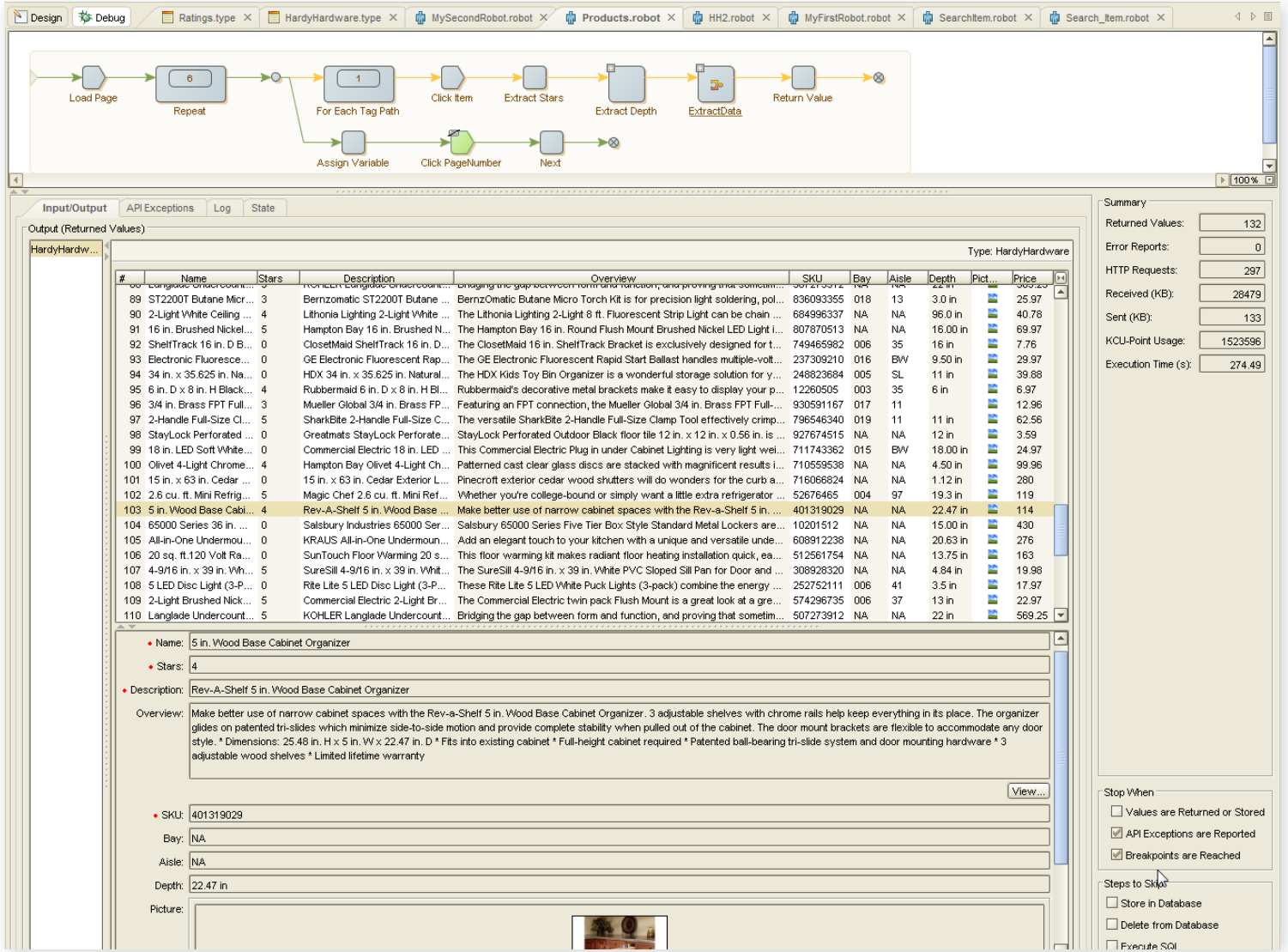
- And finally, add a "Next" step to return to the "Repeat."



- Let's review: Our robot directly loads the Products page and then loops through each item extracting the rating, the assembled depth and other information we want.
- When data for the last item on a page has been extracted, our robot automatically goes to the next alternative (branch) where it increments the page number by 1, clicks on the page number and bounces back to the repeat step where the extraction loop for that page is executed.
- This process continues until there are no more pages, and then the loop is broken.

Testing in Debug Mode

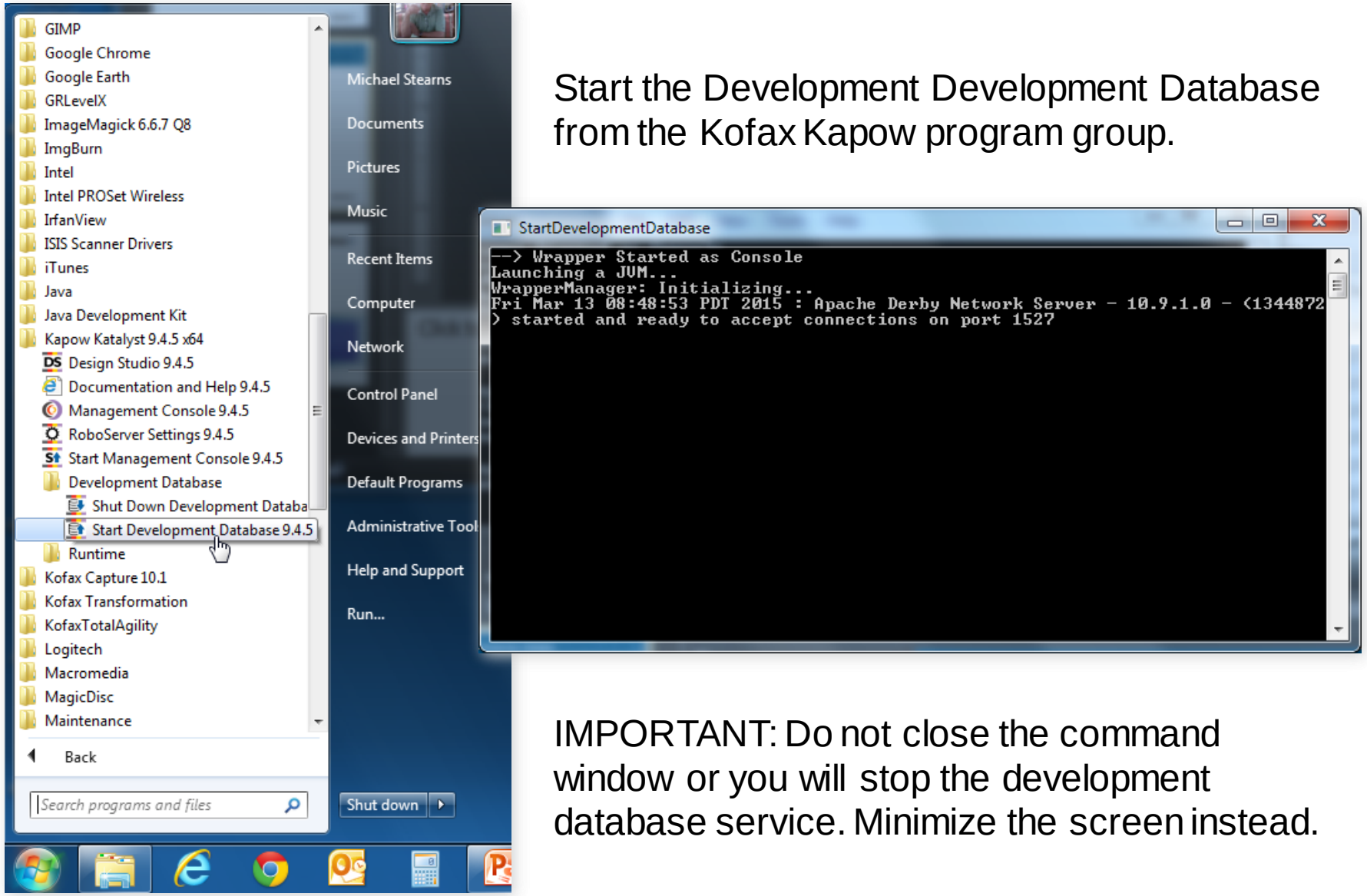
- Testing in Debug mode reveals our Robot is doing everything we designed it to do!



The Development Database – A Quick Review

- Kofax Kapow includes an Apache Derby development database.
- This database is included for learning and testing purposes only and should not be used in live production.
- Tables and columns must be created in the database to interact with your Robots. Fortunately, Design Studio makes this very easy to do.
- To make the database available, it must be started first. This can be done from the Kofax Kapow Program Group or from a command line.

Starting the Development Database

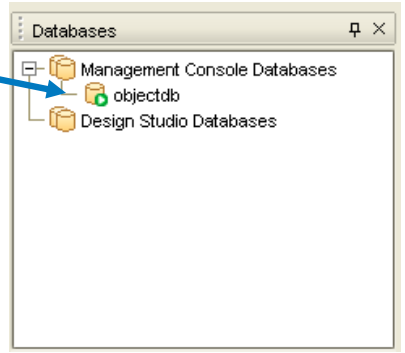


Start the Development Development Database from the Kofax Kapow program group.

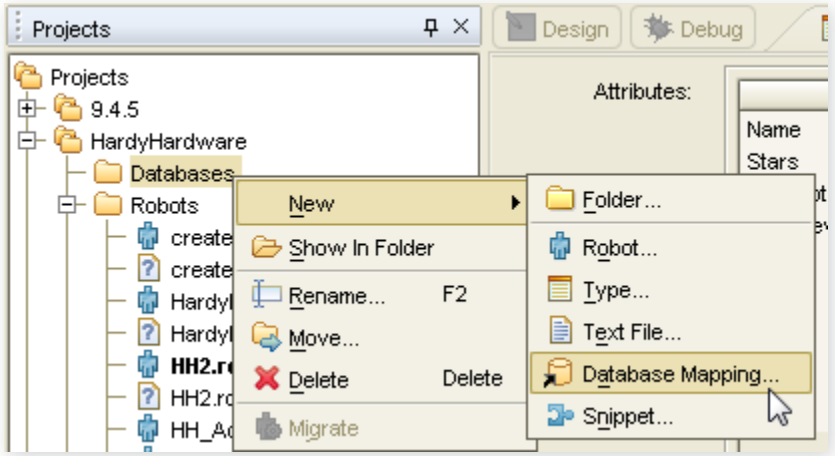
IMPORTANT: Do not close the command window or you will stop the development database service. Minimize the screen instead.

Creating Database Table

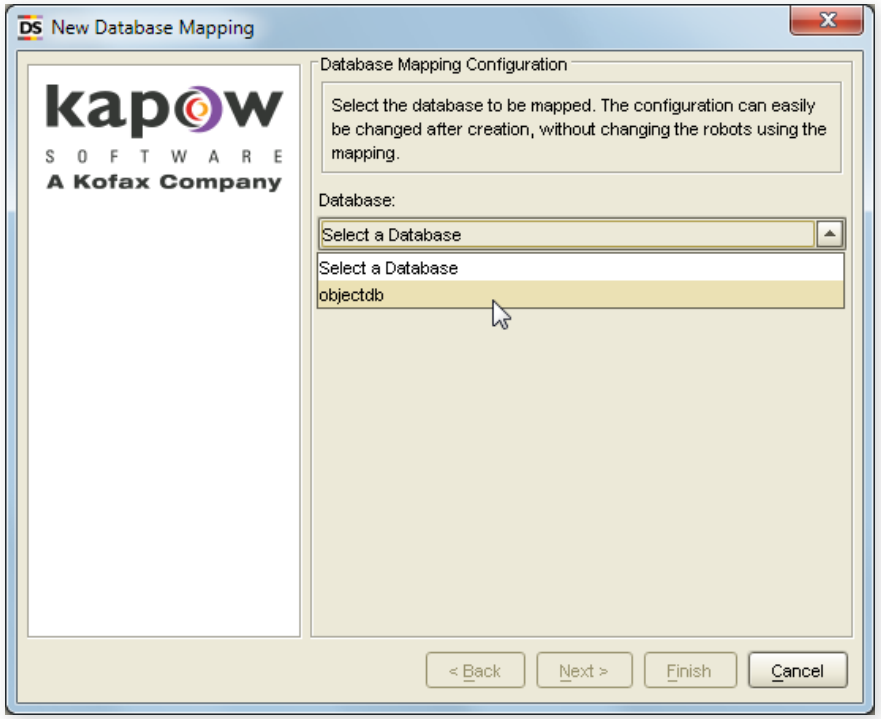
Once the database is available, the icon turns green.



1. Create new database mapping

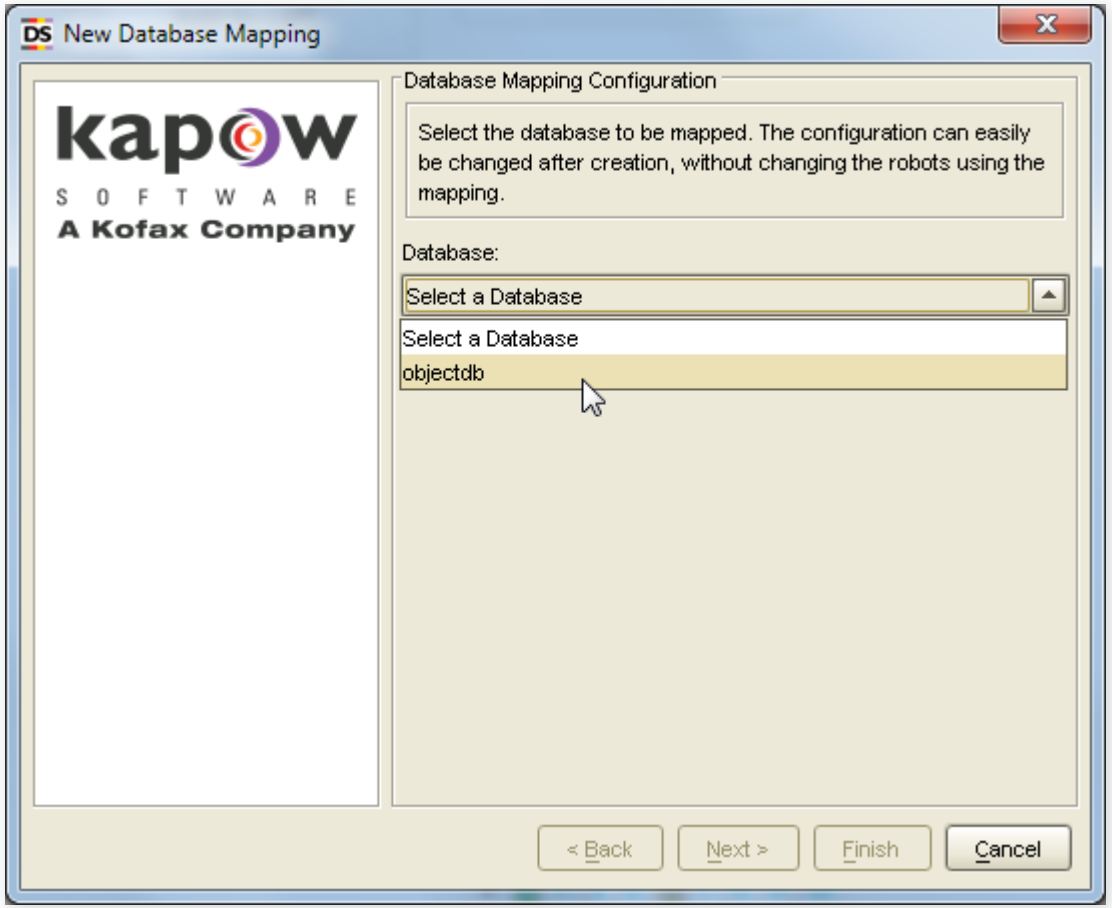


2. Select database

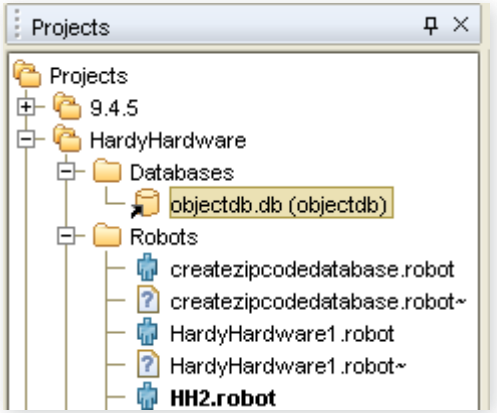


Creating a Database Table (cont.)

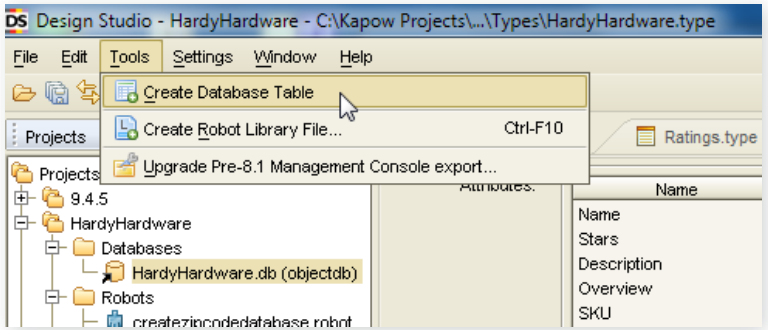
3. Select a Name for the Database Mapping



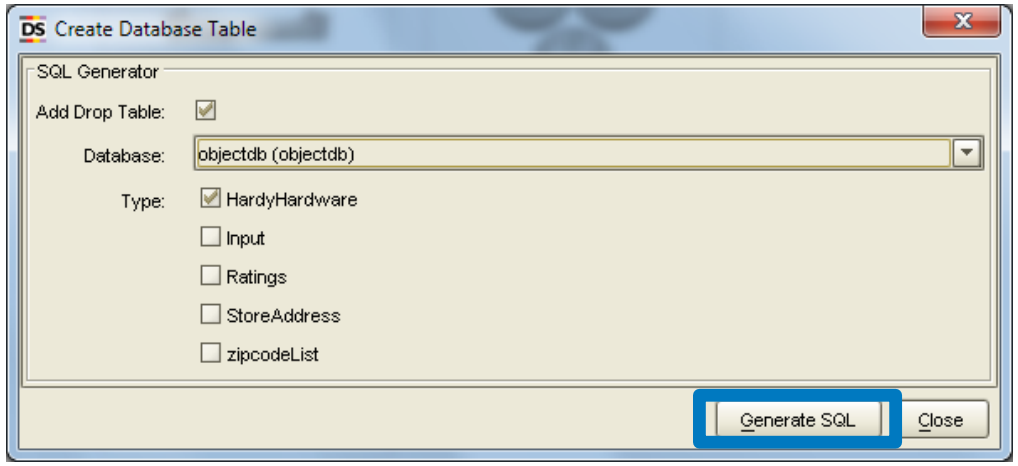
4. Database has been added

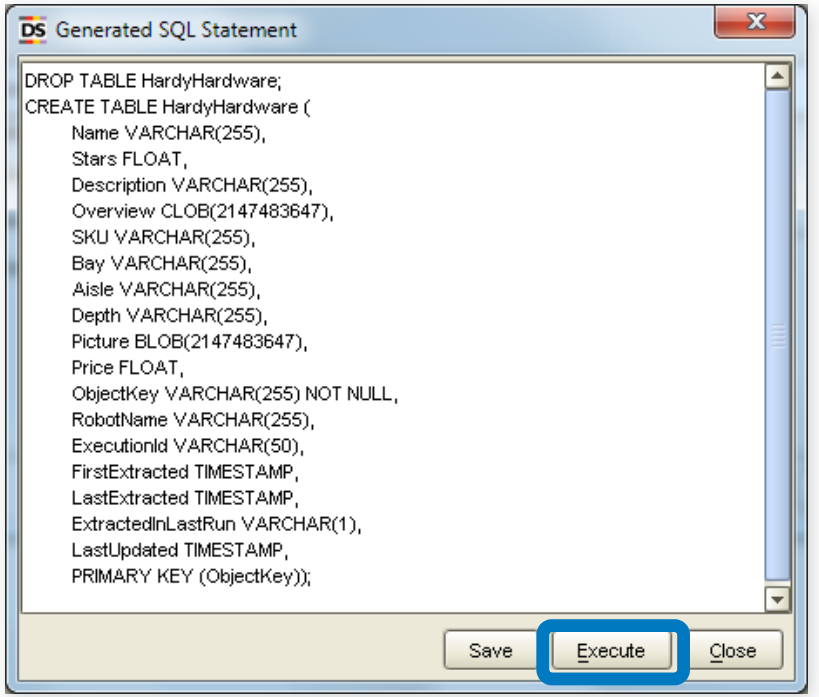


Creating Columns in your Database Table

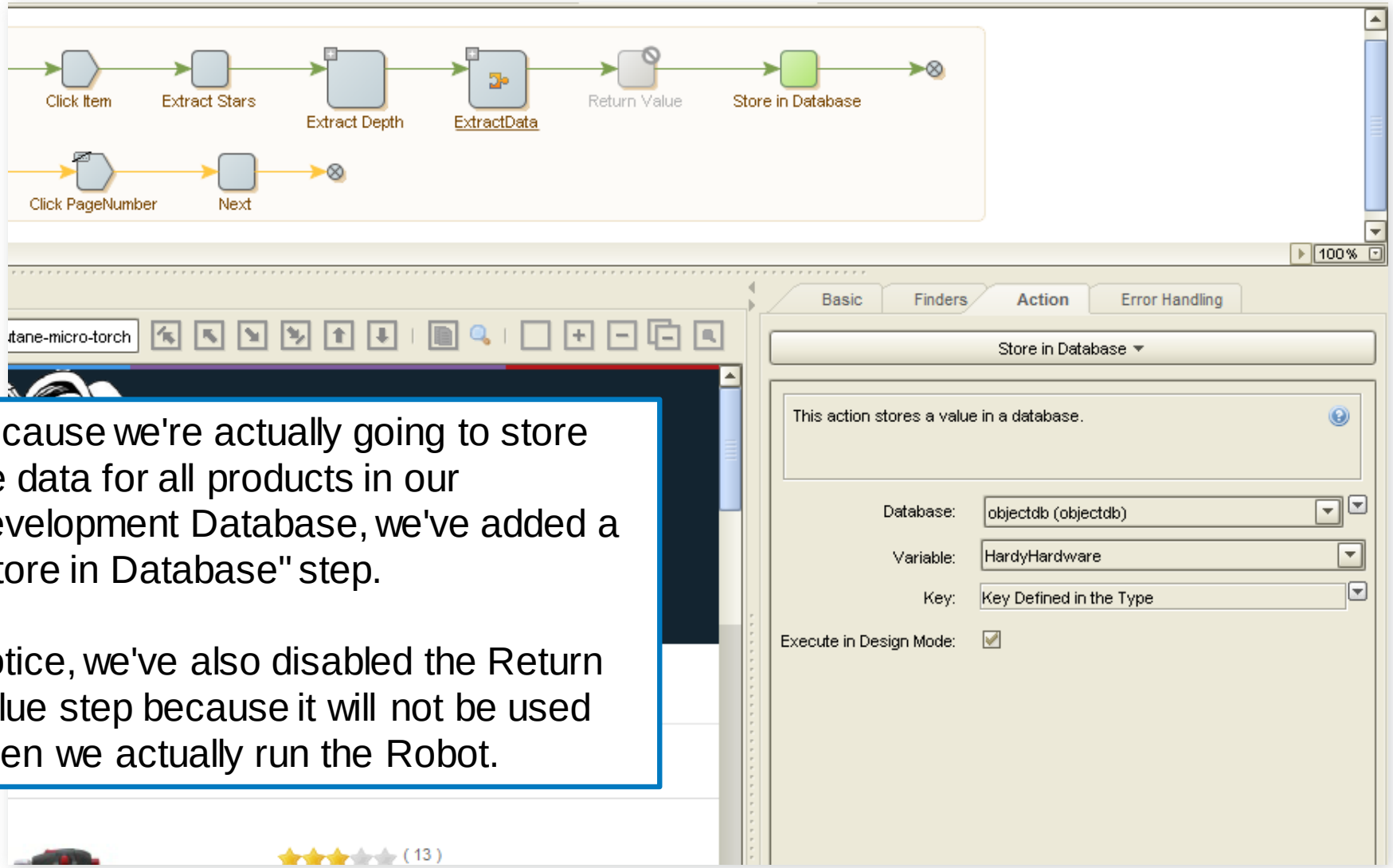


The "Tools" menu in Design Studio has the ability to create a SQL statement that will generate a table with all required columns, SQL data types, etc.





Adding an Store in Database Step

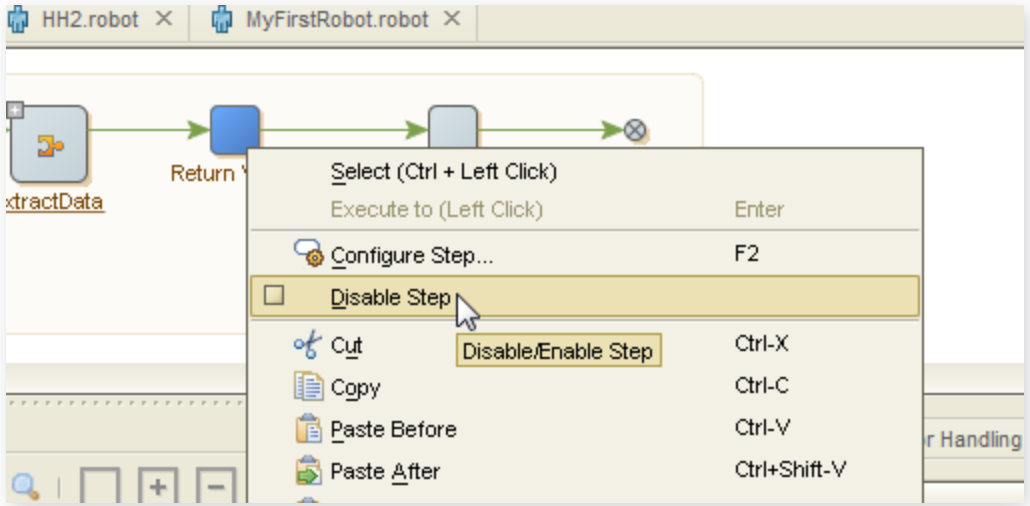


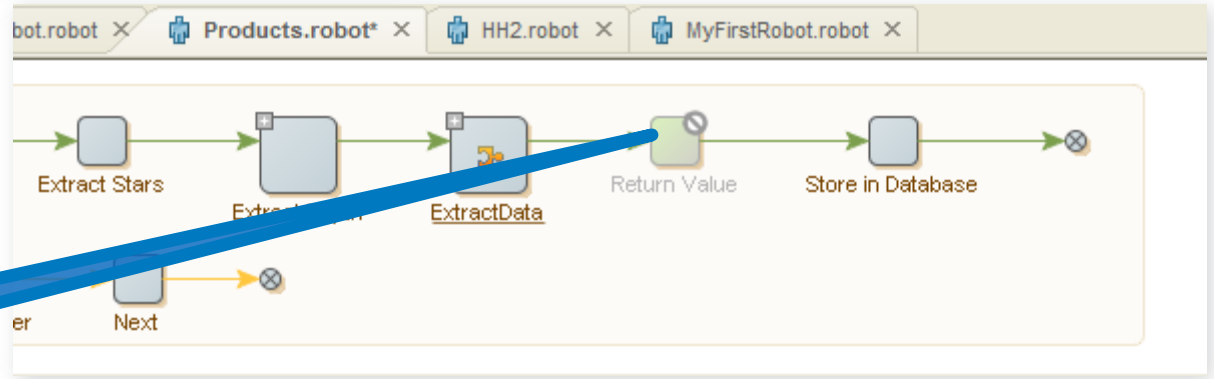
Because we're actually going to store the data for all products in our Development Database, we've added a "Store in Database" step.

Notice, we've also disabled the Return Value step because it will not be used when we actually run the Robot.

Disabling Unnecessary Steps

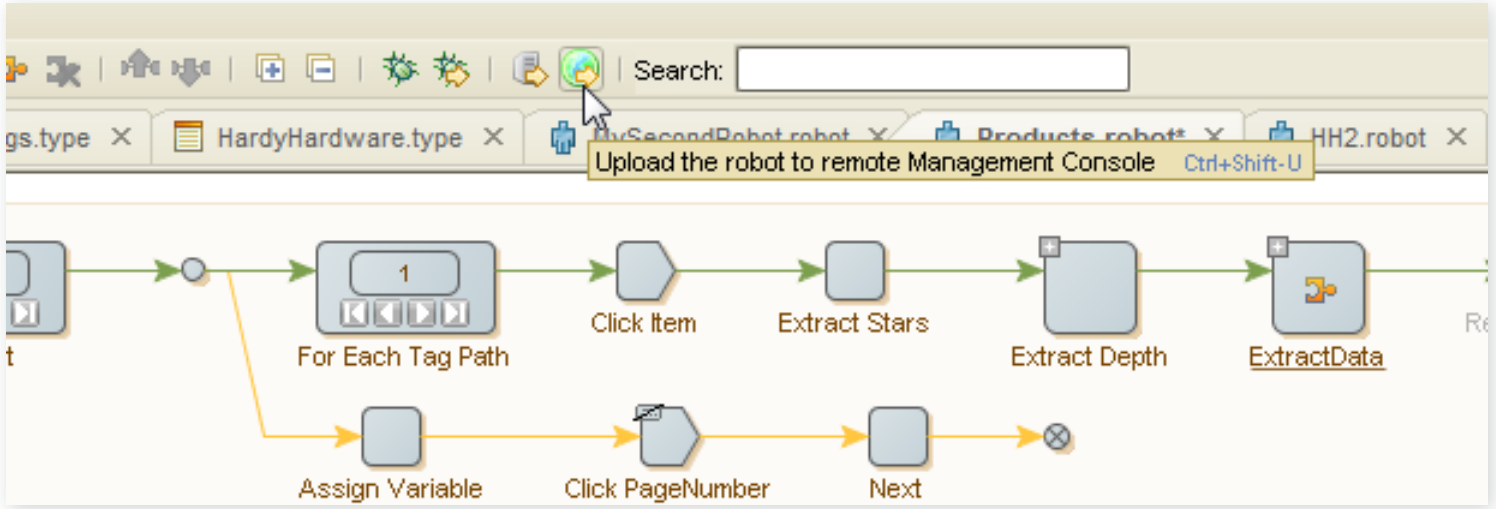
- To disable a step, simply right mouse-click on it and select "Disable Step" from the context menu. To re-enable it, right mouse-click and select Enable step. It's that easy!





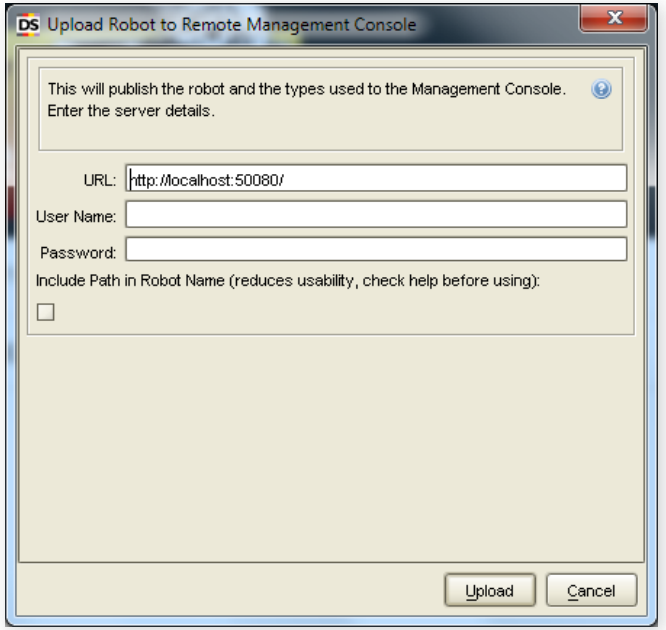
Step disabled

Uploading Robot to the Management Console



Think of uploading a Robot as publishing it…that is making it available for use.

You may Upload to a Local or to a Remote Management Console. In our example for this training class, it's simply "localhost" (our local machine).



KOFAX

Demo & Lab

Advanced Repeat/Next Loops