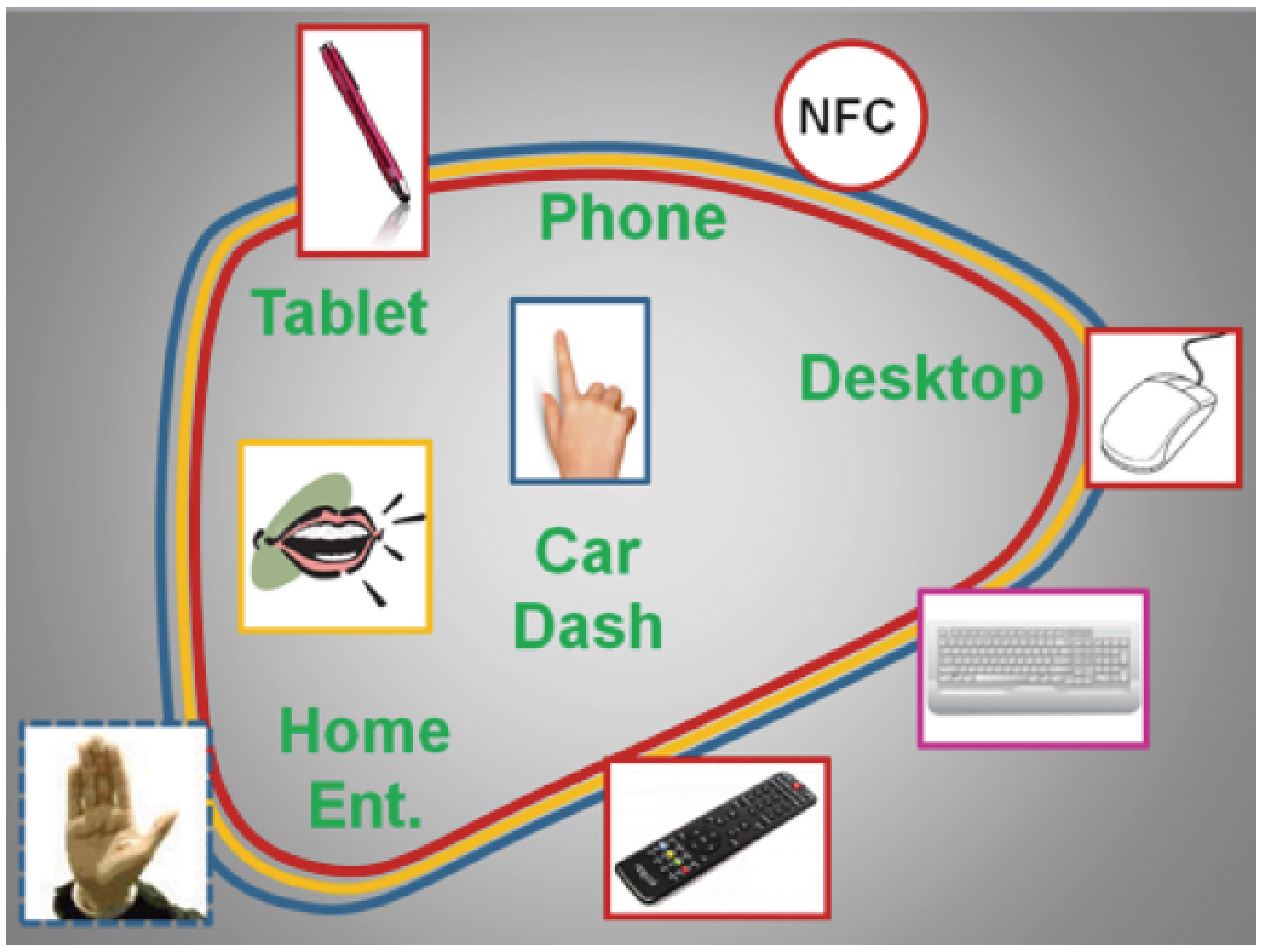
Employing a flexible interaction language scheme with User Defined Variables

There is an emerging environment where software applications need to support every computing device, every input device, and every human interaction at the same time.

One of the few benefits of getting old is that you gain some historical perspective on technological advancements. Around 1989, the mouse took over the keyboard as the main interaction device for most mainstream consumer PC users. An interesting aspect for User Assistance professionals was that it was necessary to explicitly explain how to work with a mouse instead of a keyboard in our user documentation. For example, 'Position the mouse cursor in the desired place, then press the left mouse button once'. By 1992 most people were familiar with the Graphical User Interface (GUI) and mouse and it wasn't necessary to be explicit with our instructions.

Fast forward twenty years and we find ourselves in a situation that is similar, but even more challenging. With the introduction of the iPhone and capacitive screens in 2007, the touch User Interface (UI) became practical and popular. After that, the voice interface arrived with Google Voice and Siri. Home entertainment systems are now being developed with support for hand gestures and the use of multi-purpose remote controls. The stylus is still useful for artists and others. Near field communication has its own form of interactions for completing transactions. And don't forget the reliable old keyboard.

There is an emerging environment where software applications need to support every computing device, every input device, and every human interaction at the same time. This means the majority of mainstream customers are going to need help from us to learn the language of touch, hand gestures, voice, and more.



The growing family of software interaction types

When it comes to writing procedures we all need to make an important decision, do we want to use a generic set of verbs that support all the different types of interactions designed into our software or do we want to customize the content for specific types of interactions.

If you decide to customize verbs for multiple interaction types, then you definitely want to take advantage of RoboHelp's User Defined Variables (UDVs).

Using a generic set of verbs definitely simplifies the development process. For example, you could use the word 'select' to represent choosing an item in the user interface. 'Select' can apply to keyboard, mouse or touch interfaces. But what if your customer doesn't know what type of interaction to use for selecting something. You may want to explicitly describe it in terms of the interaction type. Also, if you have standardized on using the language of the mouse—click, double-click, you will need to make adjustments for touch and other interactions.

The following examples show three alternate interaction verbs (generic)(mouse)(touch) for three different commands.

- To change the widget setting, (select)(click)(tap) Preferences.
- (Scroll)(Click the scroll bar)(Flick) to view additional items.
- (Move)(Click and drag)(Press and slide) the unused items to the trash.

If you decide to customize verbs for multiple interaction types, then you definitely want to take advantage of RoboHelp's User Defined Variables (UDVs). Adding UDVs to your procedure steps automates the process of adjusting verb choices for different interaction types. Even if you are currently using only generic verbs, you may want to use UDVs. That way if you change your approach in the future, it will be easy to add support for multiple interaction types.

UDVs have many uses in RoboHelp projects. You may already be using them for brand names or user interface terms. The rest of this article describes how to use UDVs for specifying interaction types.

There are three major steps to work with UDVs in RoboHelp:

1. Define your UDVs
2. Add UDVs to your topics
3. Assign UDV sets to screen profiles

Define your UDVs

Your user defined variables will be based on the interaction types you want to support and the language choices you choose to describe those interactions. Table 1 is an expanded version of the three command examples in the previous section. It shows the three commands and the associated language choices for five different interaction types. The Variable Name column shows the unique strings that have been associated with each command.

Command Language Interaction Table

Variable Name	Interaction Type and Values				
	Generic	Mouse	Touch	Hand Gesture	Voice
Command_ Select	select	Click	tap	✋ (hover)	say "select"
Command_ Scroll	scroll	click the scroll bar	Flick	(swipe)	say "scroll"
Command_ Move	move	click and drag	press and slide	(grab and position)	Say "move"

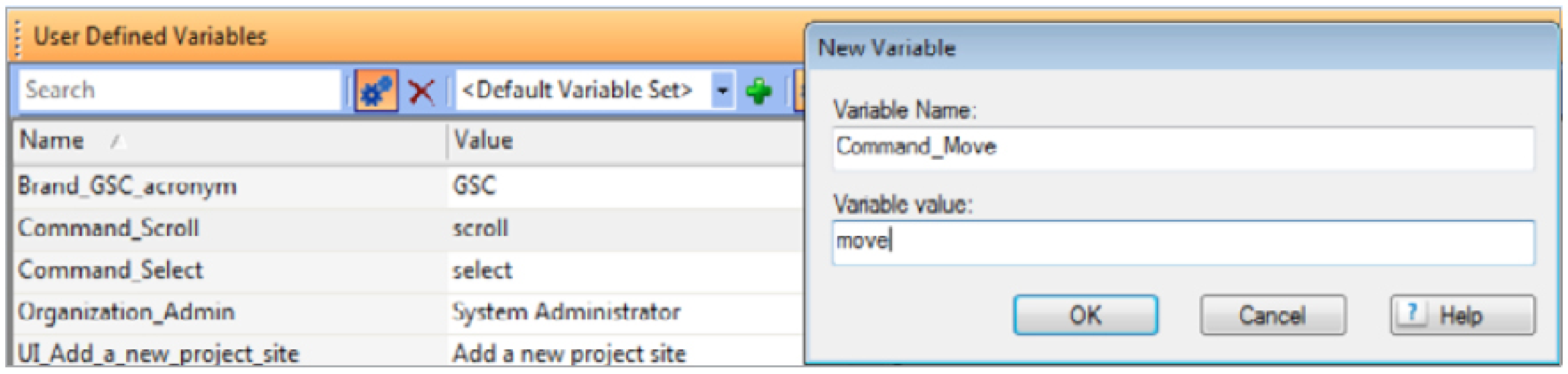
Table 1

Let's use the information from Table 1 with RoboHelp.

From the View menu in RoboHelp, select Pods, then select User Defined Variables. The User Defined Variables pod is where you will enter the word choices for your various interaction types. The list box at the top right of the pod should show <Default Variable Set>. The default variable set is a place to apply your generic command values.

This section of steps is a one-time process...once you have your UDVs in place, you can use them at any point in your topic authoring process.

Click the 'gear' icon in the pod tool bar. The New Variable box appears. Type the variable name and value for the first command: 'Command_Select' and 'select' respectively. Click OK. The new variable appears in the list of UDVs. Repeat for the other two other generic commands and values.



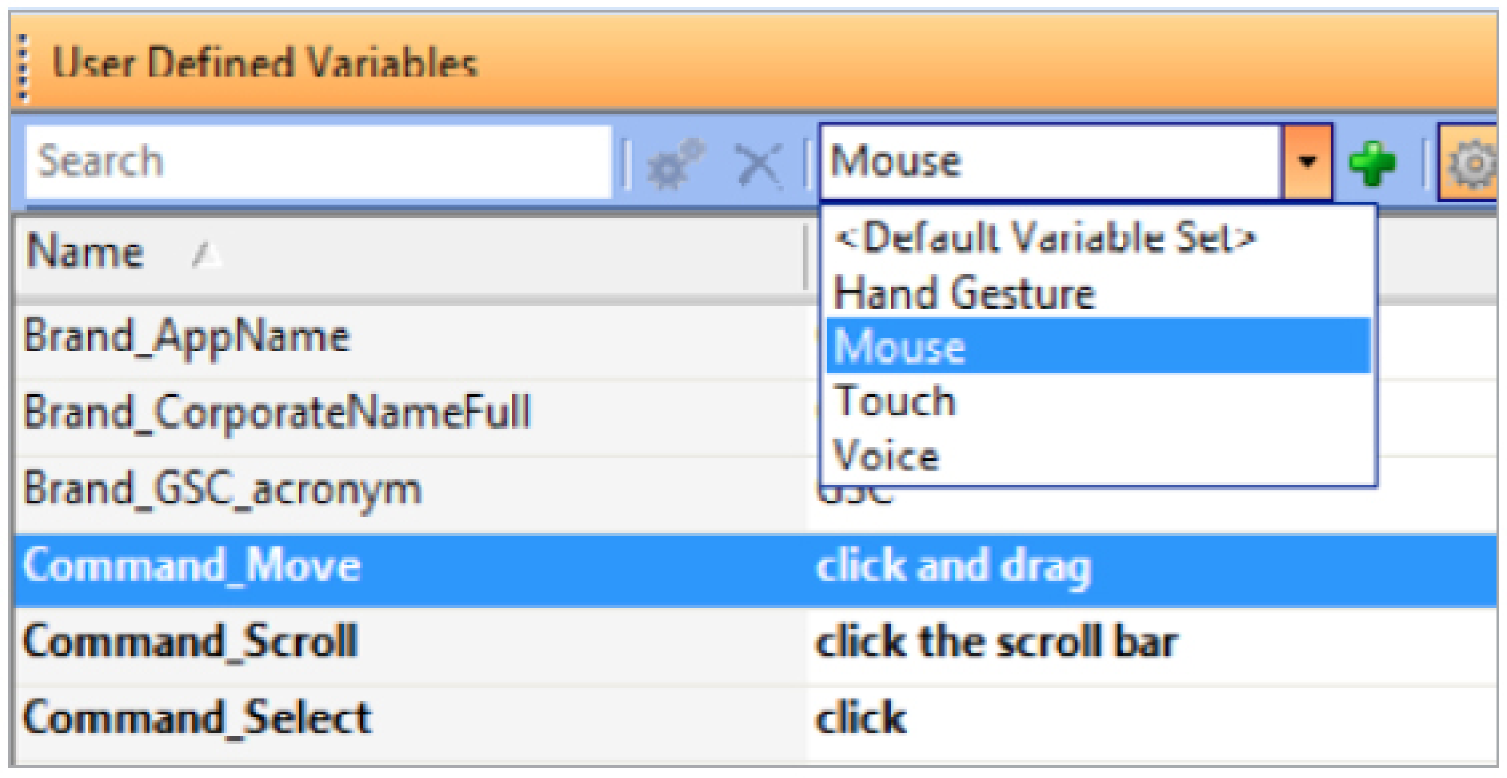
Name	Value
Brand_GSC_acronym	GSC
Command_Scroll	scroll
Command_Select	select
Organization_Admin	System Administrator
UI_Add_a_new_project_site	Add a new project site

Variable Name: Command_Move

Variable value: move

Next, create variable sets for each of the remaining four interaction types. Click the green plus-sign. In the Variable Set box, click Add. Type 'Mouse'. Use the Add button again for Touch, Hand Gesture, and Voice. Then click OK. The list box showing the <Default Variable Set> should now be accompanied by the other four interaction types.

Now you need to populate each new variable set with the associated language choices. From the variable set list box, select 'Mouse'. Notice that duplicates of all the default Command variables have automatically been created. Right click on 'Command_Select', then click 'Edit'. Type 'Click'. Repeat for the other two commands and their mouse values.



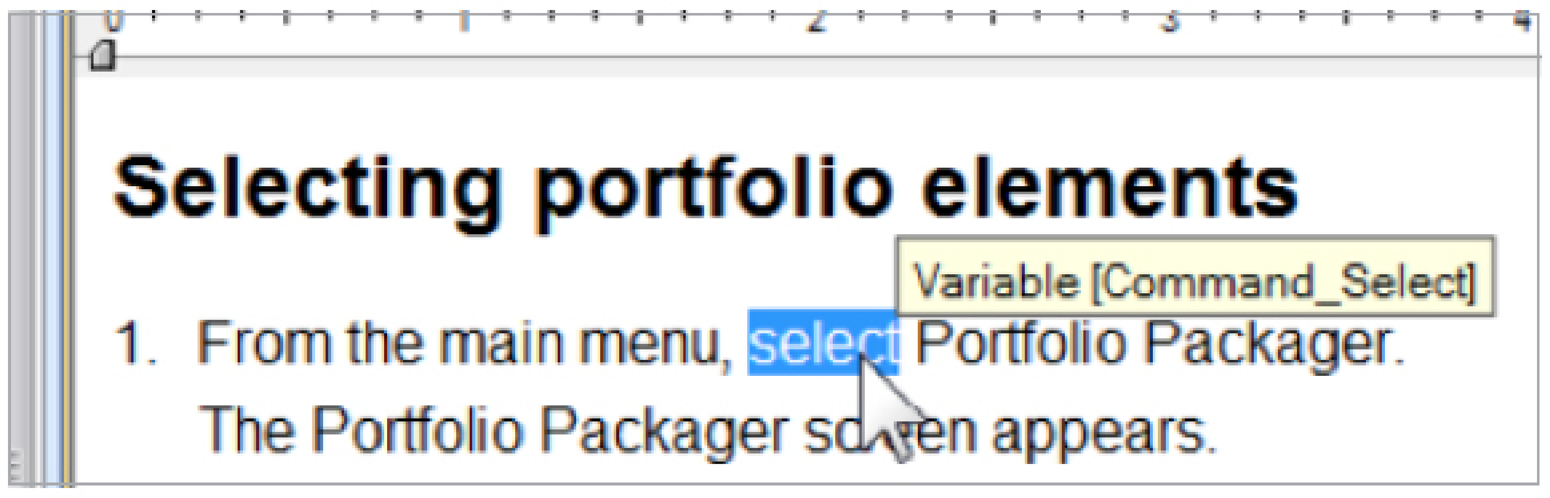
Name	Value
Brand_AppName	
Brand_CorporateNameFull	
Brand_GSC_acronym	GSC
Command_Move	click and drag
Command_Scroll	click the scroll bar
Command_Select	click

Finally, populate the Touch, Hand Gesture, and Voice variable sets. This section of steps might sound like a lot of work, but it is a one-time process. From this point on you will just be using the UDVs in your topics.

Add the UDVs to your topics

Once you have your UDVs in place, you can use them at any point in your topic authoring process. Instead of typing an interaction verb into a topic, you will add the appropriate UDV.

In RoboHelp, from the Insert menu select 'User Defined Variable'. The 'Insert User Defined Variables' box appears. Select the appropriate UDV from the list. In the Figure below, the default (generic) value of 'Select' have been inserted. Clicking on any part of the word highlights the whole word and a tool tip displays the name of the variable. These UDV objects will be automatically transformed when you generate your RoboHelp single-source output.



Selecting portfolio elements

Variable [Command_Select]

1. From the main menu, select Portfolio Packager.
   The Portfolio Packager screen appears.

...to change the values...all you need to do is select the desired variable set and then edit the desired value.

Continue to use this procedure throughout your authoring process.

You may find that at some point you need to change the values of your language choices. All you need to do is select the desired variable set and then edit the desired value. The change will automatically be reflected in your next single source build.

Assign UDV sets to screen profiles

The last part of the process is to assign your sets of interactions to the appropriate single-source outputs. If you are supporting multiple interaction types it is likely that you will be using multiscreen HTML5 as your single-source layout. With the multiscreen HTML5 layout you will have created screen profiles for different devices or categories of devices. One of the properties you can assign to a screen profile is the use of a particular UDV set.

In the Single Source Layout pod, right click on 'Multiscreen HTML' and select 'Properties'. In the Settings box, expand the 'Screen Profiles' section to display the profiles you will have already set up. In the example below there is a Screen Profile for phones using the Android operating system. Click on 'Phone_Android' to display the customizations for this device.

Click on the 'Variables Set' box. This displays the list of variable sets you created earlier in the process. For this example, select 'Touch'. If you have additional multiscreen profiles, you can specify the variable set for each one from the same 'Setting' dialog box.

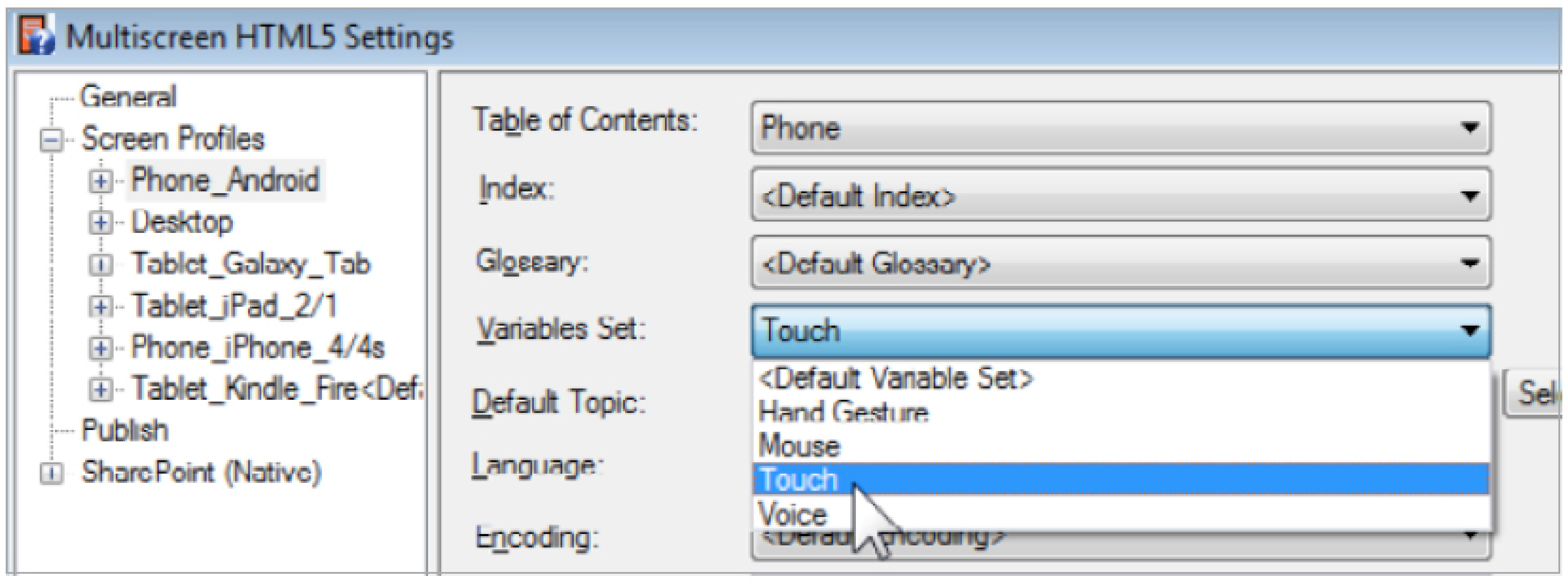

Multiscreen HTML5 Settings

Table of Contents: Phone
Index: <Default Index>
Glossary: <Default Glossary>
Variables Set: Touch

Click 'Save and Generate'. This builds the output for each of your multiscreen profiles. To view the result of a particular output, right click on 'Multiscreen HTML', select 'View with Screen Profile', then select the profile. Navigate to topics that contain interaction language and you should find that the appropriate words have been automatically rendered.

You can specify variable sets for other single source layouts like WebHelp and HMTL Help. In that case you would need to manually change the variable set for different builds.

Tips

Here are a few additional tips that might help with your organization.

Using a variable name prefix

In the UDV examples shown above, I used 'Command' as a prefix in the variable name for each command. This is done to keep all of the command UDVs grouped together in the alphabetized list. It makes it easier to find what I'm looking for. You can use any type of prefix you like. Or none at all.

Supporting capitalization

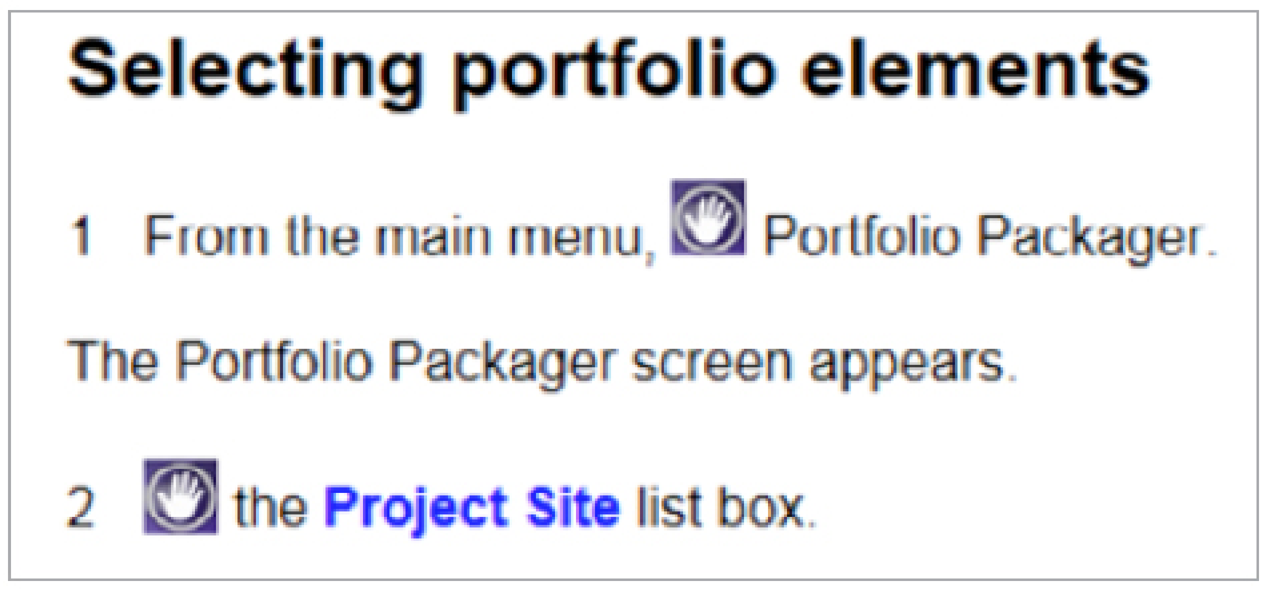
Depending on your style of writing procedures, you may have the need to display your command verbs with initial lower case and also initial caps. In that case, you might want to create pairs of command UDVs. For example,

Variable Name	Generic	Mouse	Touch
Command_Select	Select	Click	Tap
Command_Select_lc	Select	Click	Tap

Using images

For hand gesture commands, you may find an icon image to work better than a term.

For hand gesture commands, you may find an icon image to work better than a term. For example, the Xbox connect command to select something is 'hover'. They also use the image. UDVs support the inclusion of images if they are expressed in HTML. For an example, the HTML code for an image named 'hover_22.png' which resides in the RoboHelp 'images' folder would be <img alt="" src="images/hover_22.png" style="border: none;" border="0" />. Paste this string into the Value field for the 'Command_Select' variable in the 'Hand Gesture' variable set. The resulting output for a profile specifying hand gestures would look as shown in the following image.


Selecting portfolio elements

1 From the main menu, Portfolio Packager.

The Portfolio Packager screen appears.

2 the **Project Site** list box.

Next

The next article will discuss the use of Conditional Text to adjust the topic density for different output targets like phones and tablets.

Biography

Joe Welinske is the President of WritersUA.

Joe Welinske is the president and founder of WritersUA—a company devoted to providing training and information to User Assistance professionals. The WritersUA Conference draws hundreds of attendees each year from around the world to share the latest in User Assistance design and implementation. The free content on the WritersUA website attracts over 20,000 visitors each month. Joe most recently published 'Developing User Assistance for Mobile Apps'. He is currently teaching courses at Bellevue College and UC Silicon Valley. Joe received a B.S. in Industrial Engineering from the University of Illinois in 1981, and an M.S. in Adult Instructional Management from Loyola University in 1987.

Adobe Systems Incorporated
345 Park Avenue
San Jose, CA 95110-2704
USA
www.adobe.com