

DeviceAnywhere Enterprise

How to Get the Best Results with Image Matching

Copyright

Copyright © 2012 Keynote DeviceAnywhere. All Rights Reserved.

June 2011.

Notice

© 2012 Keynote DeviceAnywhere. All rights reserved.

THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT ANY EXPRESS REPRESENTATIONS OF WARRANTIES. IN ADDITION, KEYNOTE DEVICEANYWHERE, DISCLAIMS ALL IMPLIED REPRESENTATIONS AND WARRANTIES, INCLUDING ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

Test Center Enterprise 5.2 SP4

Test Center Enterprise Automation 5.2.3

Test Center Enterprise Monitoring 5.2.3

All text and figures included in this publication are the exclusive property of Keynote DeviceAnywhere, and may not be copied, reproduced, or used in any way without the express permission in writing of Keynote DeviceAnywhere. Information in this document is subject to change without notice and does not represent a commitment on the part of Keynote DeviceAnywhere. Keynote DeviceAnywhere may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents except as expressly provided in any written license agreement from Keynote DeviceAnywhere.

Mobile Complete, the Keynote DeviceAnywhere logo, DeviceAnywhere, Direct-to-Device, DeviceAnywhere Studio, Test Center Developer, Test Center Enterprise, Test Center Enterprise Interactive, Test Center Enterprise Automation, Test Center Enterprise Monitoring, DemoAnywhere, and DeviceAnywhere Portal are trademarks of Keynote DeviceAnywhere in the United States and/or other countries.

This document also contains registered trademarks, and trademarks and service marks that are owned by their respective owners. Keynote DeviceAnywhere disclaims any responsibility for specifying marks that are owned by their respective companies or organizations. If you have any comments or suggestions regarding this document, please send them by e-mail to kda-esupport@keynote.com.

Keynote DeviceAnywhere
777 Mariners Island Blvd.
San Mateo, CA 94404

Contents

About This Document.....	5
Document Outline.....	5
Typographical Conventions.....	5
Contacting Support.....	6
Additional Documentation.....	6
1 Introduction.....	7
1.1 When to Use Reference Points.....	7
1.2 Where You Can Implement Image-Based Reference Points.....	8
2 Guidelines for Implementing Image Matching.....	9
2.1 Optimal Device Settings.....	9
Device Settings.....	9
Device Wallpaper or Theme.....	9
Application Setup.....	10
Reflash, Reset, or SIM Card Change.....	11
2.2 General Script and Command Guidelines.....	11
Match Position and Starting Tolerance Levels.....	11
Wait Time.....	12
Multiple Possible Match Outcomes (Use Wait Event Command).....	13
Scrolling to Find Reference Images.....	14
2.3 Guidelines for Image Selection.....	14
Optimal Image Size.....	14
Screen Elements that Change.....	15
Dynamic Background.....	16
Transparent Icons.....	17
Repetitive Patterns.....	18
Simple Black and White or Two-Color Images.....	19
Images Defined at the Edge of a Device Screen.....	20
Grayed out UI Options.....	21
Selecting Application Icons.....	22
Looking for Android Icons.....	23
Searching for BlackBerry Icons on Non-Touchscreen Devices.....	24
Device Orientation.....	26
3 Guidelines for Advanced Settings.....	28
Additional Match and Mismatch Possibilities.....	28
Fixing Mismatches.....	29
Tolerance Sliders.....	29

About This Document

This document describes how to achieve repeatable and reliable results with image-based reference points in automated scripts in Keynote DeviceAnywhere mobile test environments.

With Test Center Enterprise products, you can create and run automated scripts to test or monitor your mobile device, application, or service. As part of your test script, you can implement *reference points* to verify the result of a sequence of device interactions.

An *image-based reference point* defines an area of a device screen as an expected result against which the outcome of a script can be verified. For instance, if your script navigates to the device idle screen, you can define an area of the idle screen as an image-based expected result. At runtime, Keynote DeviceAnywhere uses pixel-to-pixel image matching technology to match the actual device screen to the reference image for script verification.

Document Outline

This document assumes that you are familiar with the mechanics of creating image-based reference points in individual commands as well as in states—please refer to the [TCE Automation User Guide](#) for detailed, step-by-step instructions on setting reference points.

This document builds on the information contained in the *User Guide* to provide this information:

- ◆ A high-level overview of [when to use image-based reference points](#)
- ◆ Guidelines to [prepare your device for automation scripting using image-based reference points](#)
- ◆ [General script and command guidelines](#)
- ◆ Guidelines for [selecting an area of the device screen as a reference image](#)
- ◆ Guidelines on [using advanced settings to troubleshoot match failures](#)

The purpose of these guidelines is to help improve your image match ratio, repeatably generating the image matches you want and eliminating the matches you do not want, minimizing both mismatches and false positives.

Typographical Conventions

The table below describes the typographical conventions used in Keynote DeviceAnywhere documentation.

Style	Element	Examples
Blue	Links and email addresses	http://www.keynotedevicewhere.com The Document Outline section on this page describes the structure of this manual.
Bold	User interface elements such as menu items	Click My Devices in the Test Center view of Keynote DeviceAnywhere Studio.
Monospace	Commands, code output, filenames, directories	Right-click the project's <code>step groups</code> directory.
Monospace bold	User input	In a command window, type type adb kill-server .

Style	Element	Examples
<i>Italic</i>	Document titles and emphasis	Refer to the <i>Keynote DeviceAnywhere Private System Installation Guide</i> for instructions on setting up server infrastructure.

Contacting Support

If you have any comments or suggestions regarding this document, contact the Keynote DeviceAnywhere support organization for enterprise customers at kda-esupport@deviceanywhere.com. You may also send your inquiries about Keynote DeviceAnywhere product demonstrations and consulting services to this address.

Customers can find additional support information at <http://www.keynotedevicewhere.com/enterprise-support.html>.

Additional Documentation

You can find additional information in the following documents (available at <http://www.keynotedevicewhere.com/tce-documentation.html>):

- ◆ *TCE Automation User Guide*
- ◆ *TCE Automation Guide to Text Matching*

You can also access documentation from the **Help** menu in Keynote DeviceAnywhere Studio.

1 Introduction

Reference points are device conditions, screens, or output used to verify a sequence of device interactions, e.g., an application home page, login confirmation, or incoming SMS message confirmation.

Reference points can be text-, image-, or audio-based.

- ◆ *Image-based reference points* define an area of the device screen to be used for script verification.
- ◆ *Text-based reference points* use a text string from a device screen for script verification.
- ◆ *Audio-based reference points* call for your script to detect any device audio (or a specific DTMF sequence) for verification.

1.1 When to Use Reference Points

Reference points are necessary and useful for:

- ◆ Validating success or failure of your script
- ◆ Verifying that your device is in the correct state
- ◆ Verifying that a certain device event (input or output) has occurred

Periodic insertion of reference points in your script is a good practice as it verifies that a device is in the correct state for the execution of subsequent commands.

It is advisable to use reference points:

- ◆ At periodic intervals in your script, e.g., in each branch, to verify that a device is in the correct state for the execution of subsequent commands
- ◆ To verify the overall success/failure of your script or branch, i.e., verify that the correct screen was displayed
- ◆ When dynamic wait times as when loading an application or Web site require screen confirmation before proceeding
- ◆ When slow device response is slow due to network latency, requiring screen confirmation before continuing with a script

Reference points are also useful because you can trigger error messaging in the **Timeout** tab of commands that define reference points. The errors triggered (and their custom annotations) are included in test results for more complete reporting.

1.1.1 When to Use Image-Based Reference Points

Use the following guidelines when choosing which UI-based (i.e., text- or image-based) reference point to implement:

- 1 Whenever possible, use a text-based reference point first before making the decision to use an image-based reference point.

Image-based reference points use image matching technology to match the actual device screen pixel-by-pixel to the reference image, while text-based reference points use optical character recognition (OCR) technology to match the reference string to the actual device screen. As OCR technology can

match a string of text even with changes in font face and size, text matching can often be more robust than image matching.

- 2 Use an image-based reference point when text matching is not feasible, as in the following cases:
 - There is not enough contrast between the text and its background for text to be correctly extracted (recognized) from the device screen.
 - Text on the screen is subject to change, e.g., time or date information, leading to failed matches.
 - Text on the screen does not uniquely identify the screen, generating the possibility of false positives.
 - There is no text on the screen.

1.2 Where You Can Implement Image-Based Reference Points

You can create *device-specific* reference points in certain scripting commands or *device independent* reference points in *states*.

You can create script- or device-specific reference points in the following commands:

- ◆ Wait Image
- ◆ Wait Event
- ◆ Send Keys
- ◆ Navigate To

NOTE You can also specify an area of the screen for your script to touch in the Find and Touch command. While the image area does not act as a reference point, the guidelines for selecting and troubleshooting an image apply to this command as well.

Device- or script-specific reference points are called so because they cannot be reused across other scripts.

You can create a device-independent reference point in a *state* that is defined for all project devices. States consist of device-specific implementations to account for differences in interfaces. Once you have created a state, use the Wait State command to call the appropriate implementation in an action or test case. You can also simply drag a state into your action/test case script from the project directory.

The mechanism for defining a reference image in device-specific commands or device-independent states is the same.

Refer to the *TCE Automation User Guide* for detailed explanations of how to define and use text-, image-, and audio-based reference points.

2 Guidelines for Implementing Image Matching

This chapter presents general guidelines for using image-based reference points and specific guidelines for defining reference images based on characteristics of the device screen and certain device conditions.

2.1 Optimal Device Settings

Follow these device setup guidelines as you prepare for automation scripting with image-based reference points, regardless of the type of reference image chosen.

These guidelines are aimed at:

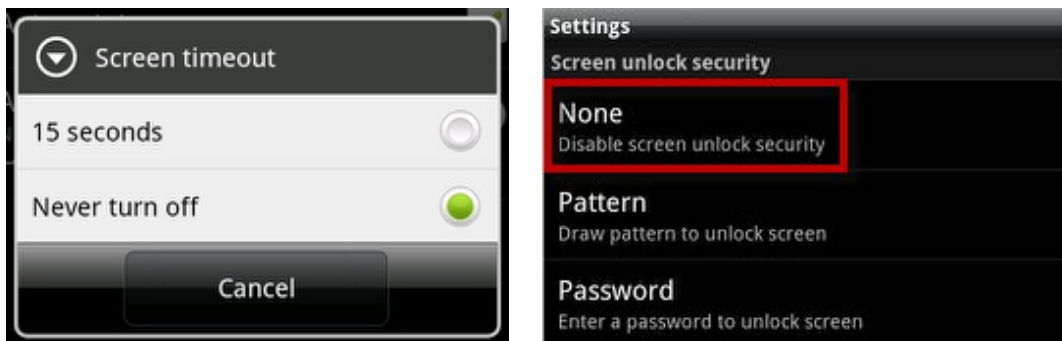
- ◆ Keeping your device as far as possible in a consistent state, avoiding changes to the theme or settings, firmware upgrades, and other changes that can interfere with script execution (including reference images)
- ◆ Minimizing unnecessary script steps and ensuring smooth and reliable script execution

Device Settings

Disable any device controls that automatically change the device screen as they can invalidate saved reference images or cause your script to fail.

- ◆ Screen timeout—turn off any backlight timer control or set it to the highest possible time so that the screen stays lit.
- ◆ Screen lock and passwords—if possible, disable or extend the maximum time available for screen timeout so that your script can be executed as written. Disable passwords and lock patterns so you do not have to script around them.

Figure 2-1 Screen Timeout and Lock Settings in Android



Device Wallpaper or Theme

Choose a device theme or wallpaper with minimum animation and color/font changes as these variations can cause image match failures. Wherever possible, choose a static, solid background.

Notice, for instance, how consecutive images of the HTC EVO home screen below vary because of the animated wallpaper that displays a live clock and swaying grass.

Figure 2-2 Screen Animation Will Cause Image Match Failures



Application Setup

Move your application icon to the top of the home screen or application tray so you do not have to create additional script steps to find and launch it. Do not make changes to the default layout and appearance of the application so you can test its usability and functionality as designed.

Figure 2-3 Move Application Icon to Home Screen for Easy Launch



Reflash, Reset, or SIM Card Change

After setting up devices for automation scripting, avoid reflashing/resetting a device or changing a SIM card in the course of your testing.

Reflashing a device, or upgrading the operating system, can often result in changes to the device UI in terms of menus and applications. Be sure to check whether the OS upgrade has impacted any stored reference images by running the command in question. Recapture reference images if you need to.

Resetting a device (factory reset) almost certainly results in changes in audio and video settings such as theme or default options. Additionally, applications, including the Keynote DeviceAnywhere Agent, are erased. You will need to redefine reference images and on software-integrated or hybrid devices, reinstall the Agent for communicating with the Ensemble Server.

You might need to *change a SIM card* on your GSM device either because you have moved to a different carrier or have switched plans on the same carrier. Although your device UI should not change, certain menu options and applications might not be available to you. Also, there might be changes in carrier name, location, and date information as well as display language. Be sure to check whether changing a SIM card has impacted any stored reference images by running the command in question. Recapture reference images if you need to.

NOTE Rebooting or power cycling a device does not impact any stored reference images.

2.2 General Script and Command Guidelines

As you begin setting up automated test or monitor scripts, follow these command guidelines:

Match Position and Starting Tolerance Levels

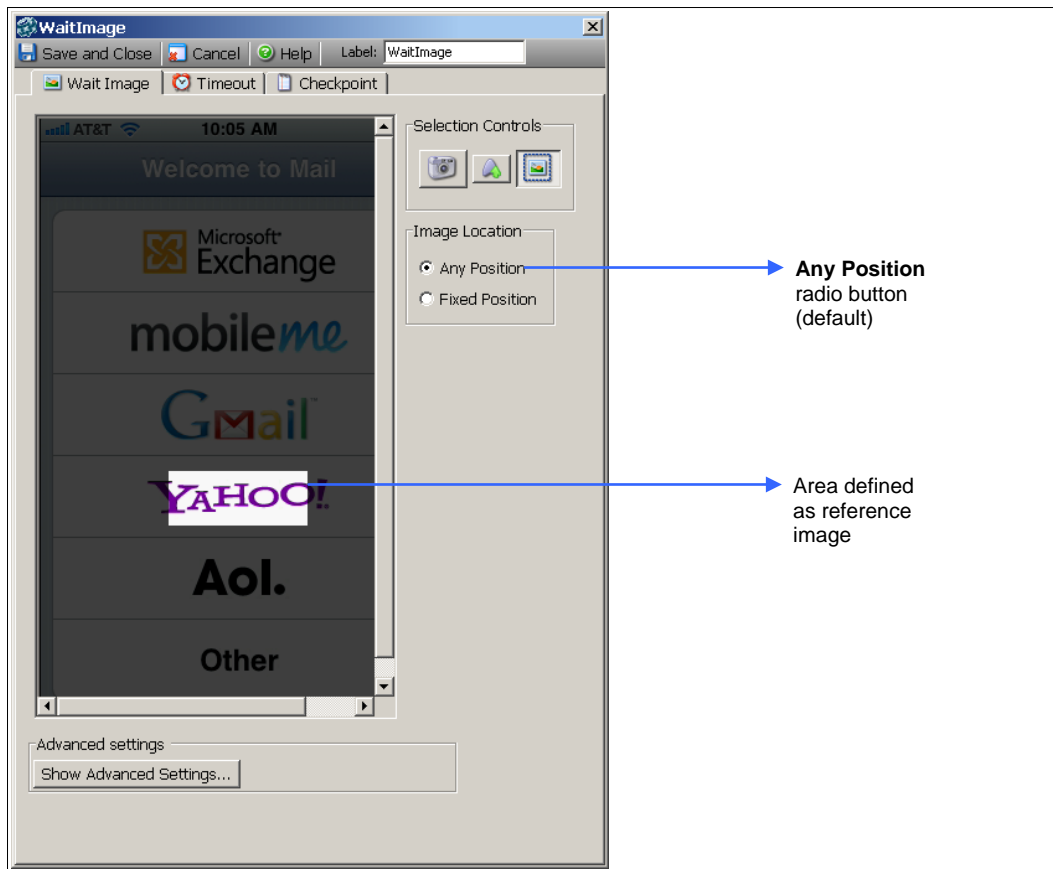
When creating a reference image in a command (e.g., Wait Image) or a state:

Leave the **Any Position** selection for **Image Location** as is. This locates the reference image at any position on the device screen (see Figure 2-4 below) and ensures that your reference image is matched even if its position on the screen changes, e.g., if your reference image is an application icon whose location on a smartphone touchscreen might vary.

Do not adjust **Advanced Settings** unless you experience match failures (see [Guidelines for Advanced Settings](#)).

The image below shows the match position setting in the Wait Image command.

Figure 2-4 Match Position



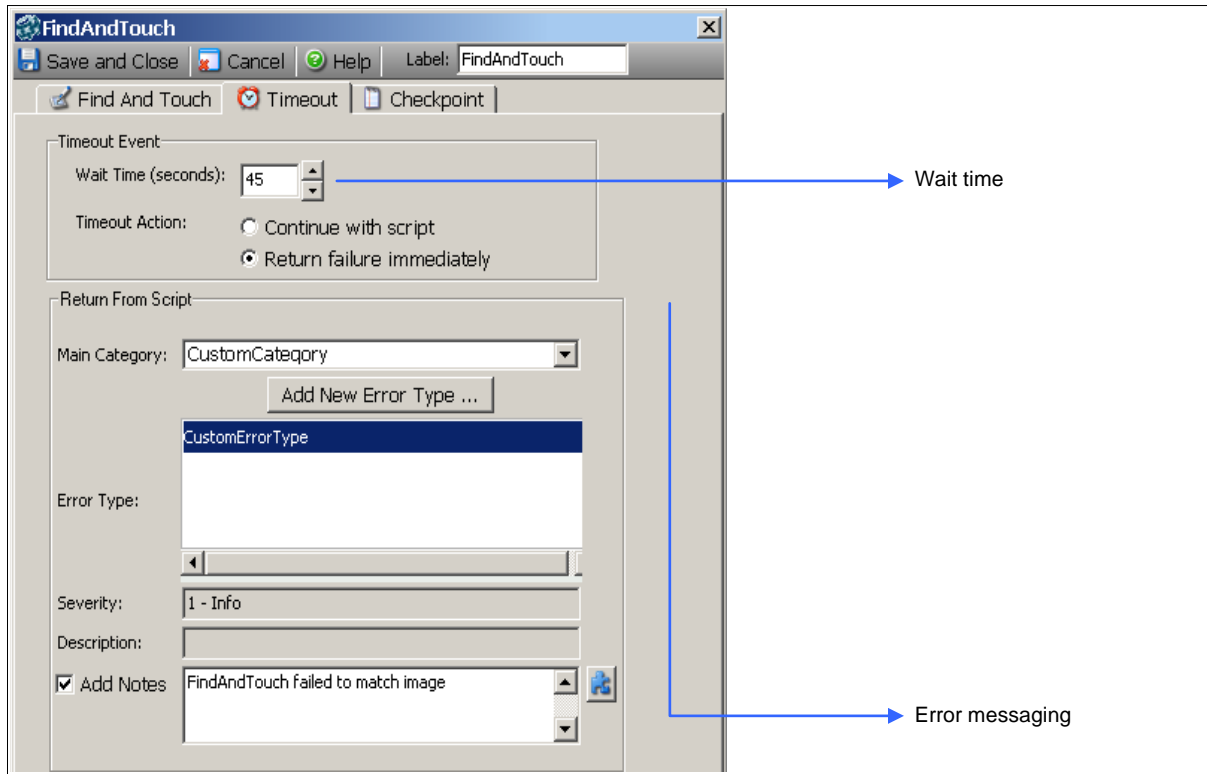
Wait Time

Provide sufficient wait time for preceding keystrokes to be performed and a reference image to be matched. A Wait command set to 2 seconds can be easily inserted after a Send Key command to allow the device some time to “catch up” with the script.

If you need to perform a swipe before your Find and Touch command can find an application icon—insert a swipe, and next, in your Find and Touch command, insert sufficient time in the **Timeout** tab for the reference image to be found. Optionally, implement error messaging in case the command fails.

NOTE You can use a Wait command to insert wait time before a Wait State command.

Figure 2-5 Wait Time in Find and Touch

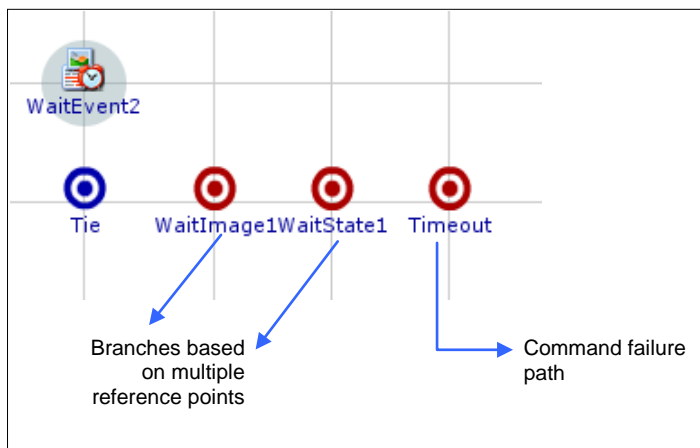


Multiple Possible Match Outcomes (Use Wait Event Command)

Use the Wait Event command in place of the Wait Image or Navigate To commands when there are multiple possible outcomes for a sequence of device interactions. The Wait Event command offers the same functionality as Wait Image with additional flexibility and robustness.

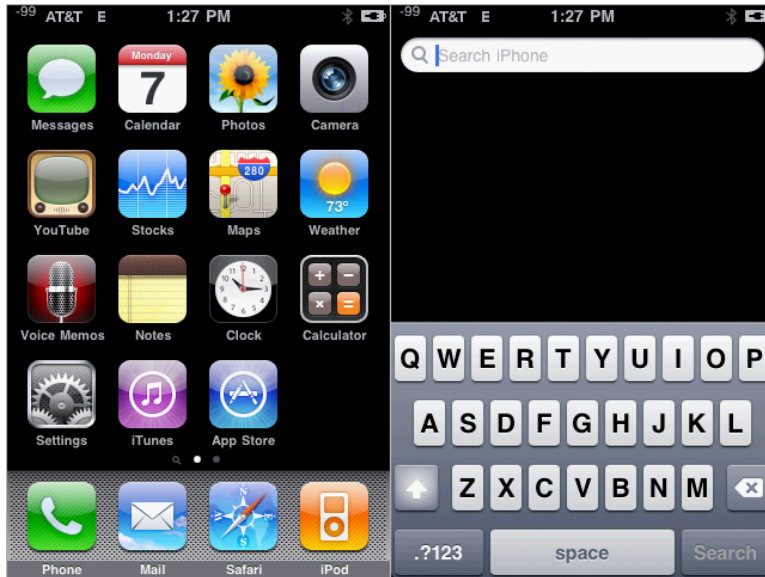
- ◆ Wait Event allows you to create branches based on multiple reference points (image, text, or audio). This is especially useful when defining reference points for multiple possible outcomes.
- ◆ Wait Event allows you to define a failure path in the Timeout branch.
- ◆ Use the Tie branch to continue with the script after a branch has been taken.

Figure 2-6 Wait Event Branches



Use the Wait Event command when a specific key sequence can have multiple outcomes. For instance pressing the Home key on an iPhone toggles between the home and main search screens (see Figure 2-7 below). You can use a Wait Event command to create a reference point for each of these screens.

Figure 2-7 Home Button Toggles Between Application and Search Screens on iPhone



Scrolling to Find Reference Images

If you need to scroll to find a reference image on a Web page (e.g., using the Navigate To command), always implement a greater number of scrolls than it takes you to find the image during script development. This allows you to factor in dynamic changes to page length that impact where your image lies. For example, if it takes you 5 scrolls to find an image from search results on eBay, implement your script with 10 scrolls to account for runtime changes to the number of search results.

2.3 Guidelines for Image Selection

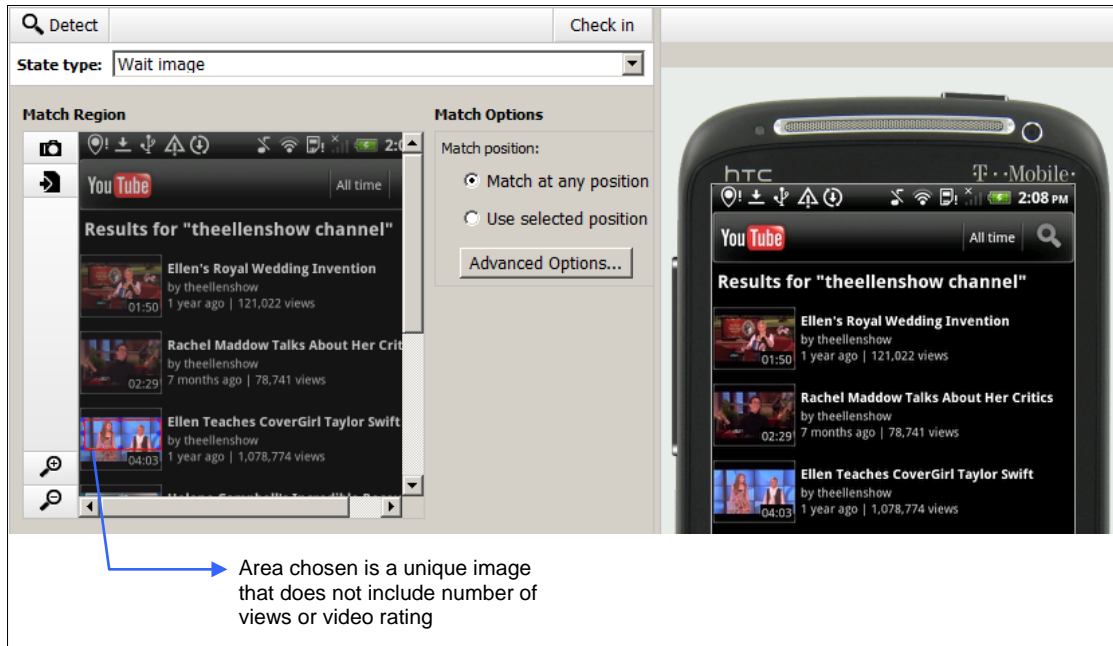
This section presents guidelines on how to choose an area of the device screen as a reference image. Reference image selection must take into account characteristics of the screen from which it is chosen.

Optimal Image Size

Choose as large an image as possible without running into possible areas of mismatch such as dynamically updated screen elements. In general, the smaller the area, especially in an image containing repeated patterns, the higher the likelihood of its being incorrectly matched to a different screen or to a different area on the same screen.

If your image area is too large, you run the risk of including screen elements that are subject to change. For instance, if you want to use a specific YouTube video to define a reference image on an Android (see Figure 2-8 below), choose an area that does not include the number of views.

Figure 2-8 Optimal Image Size



Choosing an entire screen is too large a selection and is not advisable for the reason that it might contain several elements that can change. In the image of an iPhone home screen below, areas that are likely to change are highlighted.

Figure 2-9 Changing Screen Elements on an iPhone Home Screen



Screen Elements that Change

Including screen elements that change in a reference image can cause image match failures. Examples of such elements are the signal and battery strength icons, time, date, and scroll bars that can change position.

The image below shows two versions of the same screen area. Areas of mismatch between the two versions (the time, signal, and battery indicators) are highlighted.

Figure 2-10 Screen Element Changes



The image below shows two versions of the same screen area but with different scroll bar positions. When selecting a reference image, be sure to exclude scroll bars that can change position.



Dynamic Background

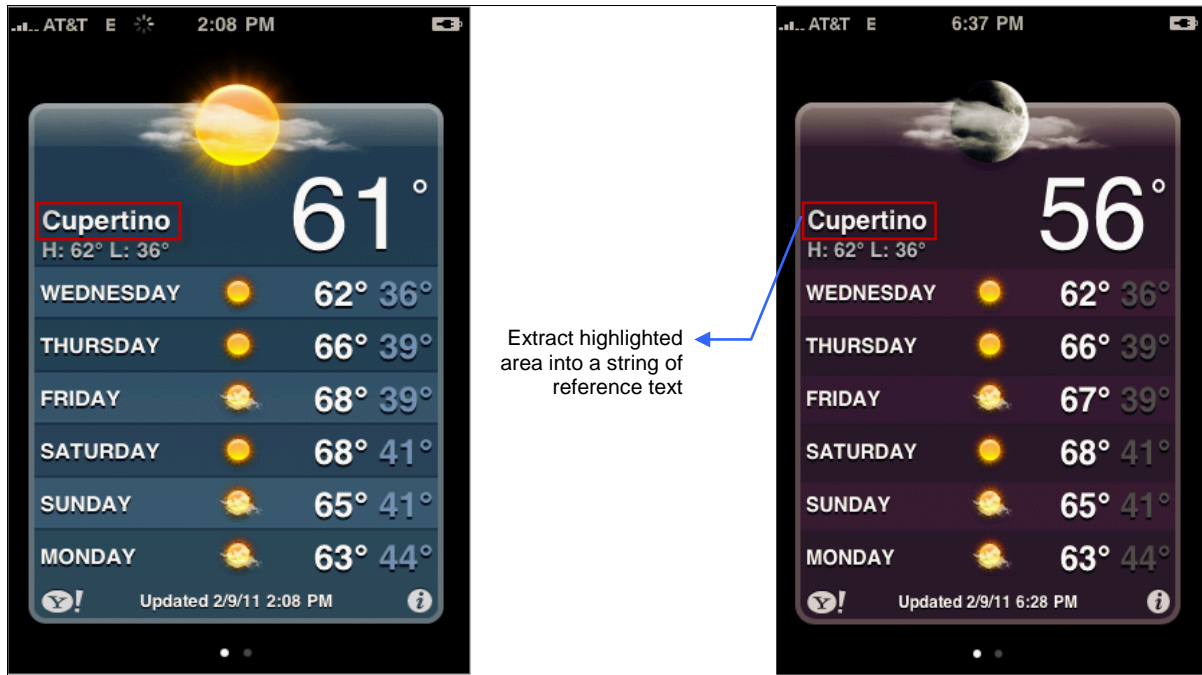
Some device screens have relatively little stable information and can change depending on certain conditions.

In such cases, your options are to:

- ◆ Turn off the background change option in your application, if possible.
- ◆ Ensure that color tolerance is at the lowest possible level to minimize the effect of a dynamic background (see [Guidelines for Troubleshooting Images](#) below).
- ◆ Only use static portions of the screen in a reference image, such as the area *inside* an icon or image.
- ◆ Match to a string of static text extracted from the screen.
- ◆ If there is no static information on a screen, try to implement a reference point either before or after that point in your script.

In some cases, it may not be possible to use image matching to validate the screen. In the iPhone weather application, the background varies according to weather and time of day for the same location. The images below show how the application changes background, images, and weather information, with the only static information being the name of the city. The screen changes so much that almost any reference image would fail. In this case, it is preferable to use the name of the city (Cupertino) to define a string of reference text to be matched.

Figure 2-11 Dynamic Background



Transparent Icons

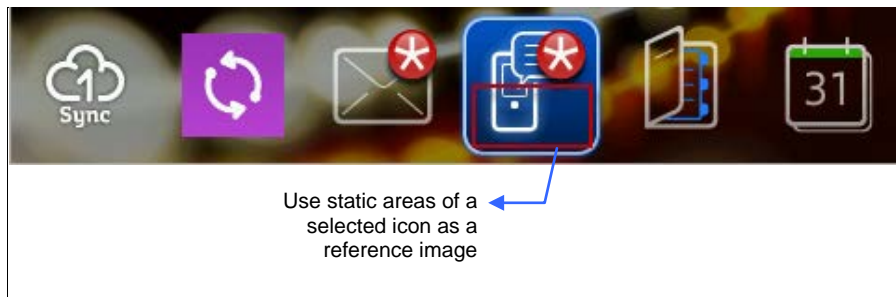
Some device screens have transparent icons on a variable background. If you choose some of the transparent background with an icon as a reference image, you might experience match failures if the background changes.

In this case too, your options are to:

- ◆ Turn off the background change option in your application, if possible.
- ◆ Ensure that color tolerance is at the lowest possible level to minimize the effect of a dynamic background (see [Guidelines for Troubleshooting Images](#) below).
- ◆ Only use static portions of the screen in a reference image, such as the area *inside* an icon or image.
- ◆ Match to a string of static text extracted from the screen.
- ◆ If there is no static information on a screen, try to implement a reference point either before or after that point in your script.

In the icon bar of a BlackBerry Bold (below), the selected icon is solid while some others are transparent. To use an icon as a reference image, navigate to it first and then choose a static area of the icon. (Ensure that your script contains commands to navigate likewise to the application icon so that it appears solid.)

Figure 2-12 Transparent BlackBerry Icons



In the example below, the browser icon on an HTC Nexus One device has a transparent background. To use the icon as a reference image, select inside the icon, making sure not to include any transparent areas.

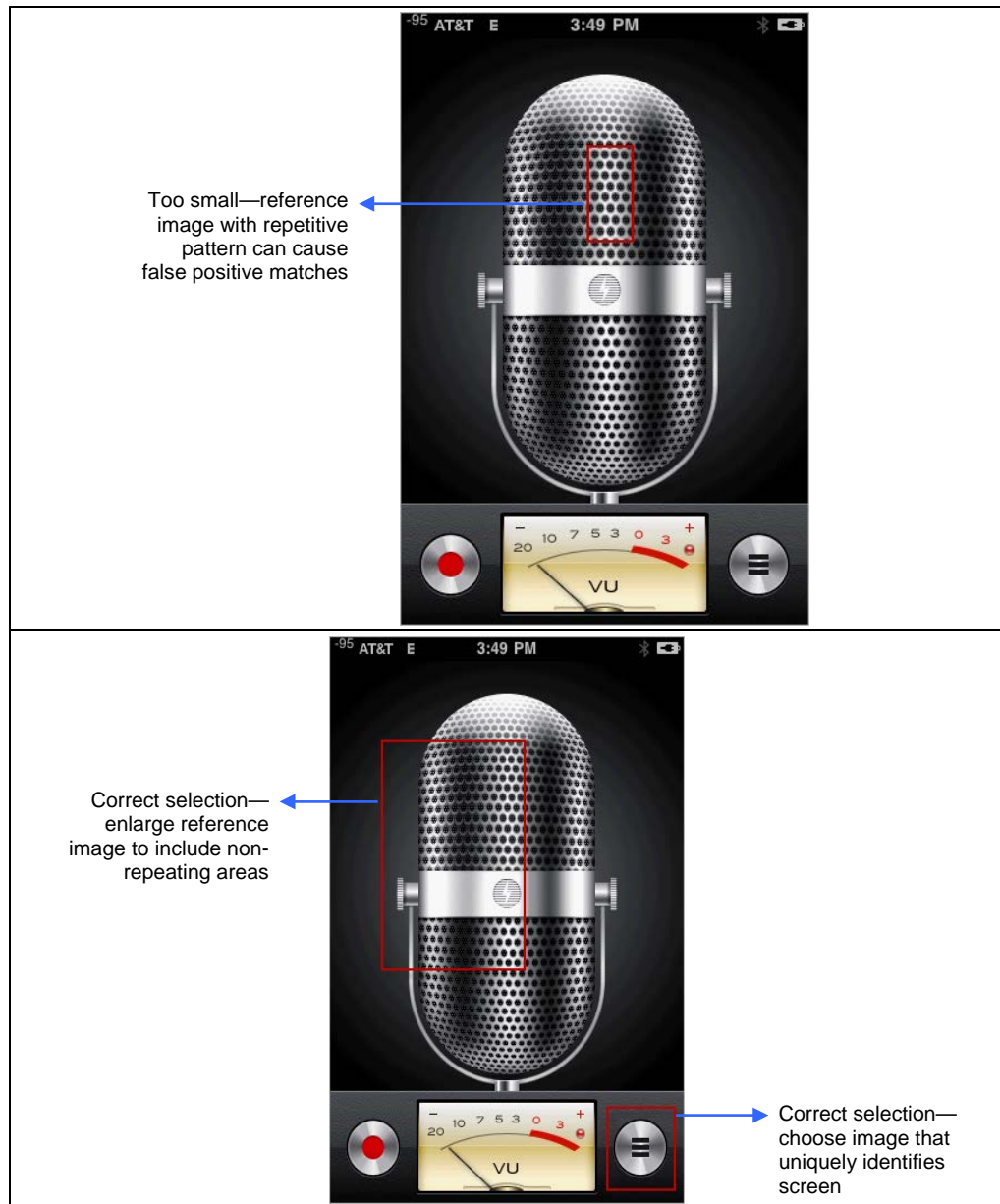
Figure 2-13 Icons with Transparent Background



Repetitive Patterns

A reference image consisting solely of repetitive patterns (especially with high position tolerance) runs the risk of being matched to a different area on the same screen or to a different screen. To avoid these false positive results, enlarge your reference image to include non-repeating areas of the screen. Or try to choose another image with no repetitive patterns.

Figure 2-14 Repetitive Patterns in a Reference Image



Simple Black and White or Two-Color Images

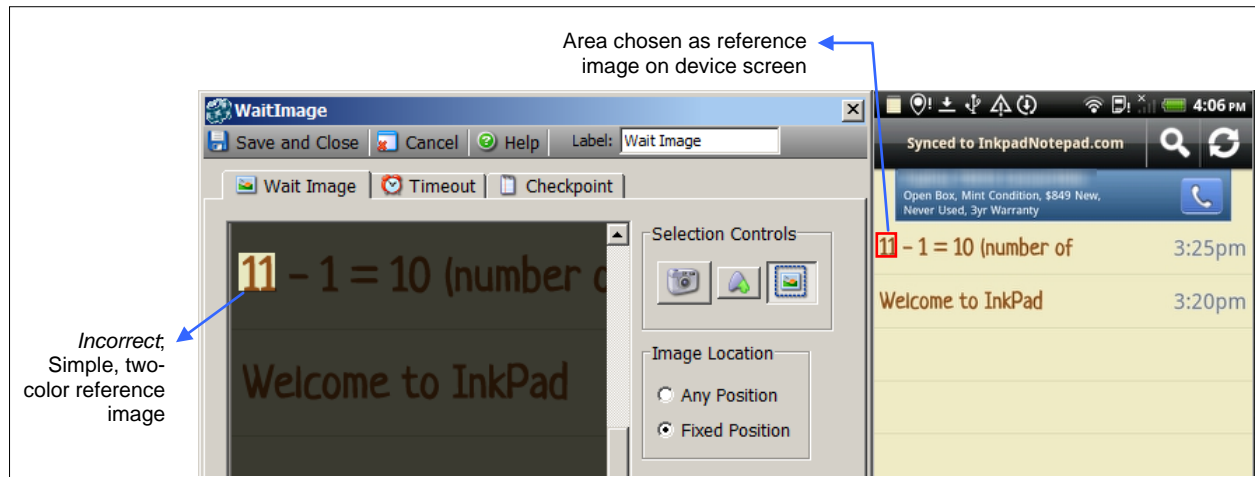
If you select a small enough region of a black and white or simple, two-color image (with high pixel and position tolerances), it runs the risk of being matched to a different area on the same screen or to a different screen. Such false positive results are especially seen when your image consists of a simple line drawing or one- or two-stroke characters.

To avoid these false positive results:

- ◆ Select a large enough area to disambiguate the image.
- ◆ If possible, select an image with more colors.
- ◆ Use low pixel and position tolerance levels (i.e., leave your sliders at the higher end of the scale).

The image below shows a reference image of a small, dichromatic area consisting of single-stroke characters.

Figure 2-15 Small, Dichromatic Reference Image with High Tolerance Levels

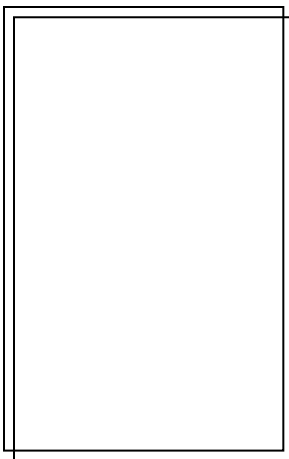


Images Defined at the Edge of a Device Screen

Avoid including the edge of the device screen in your reference image in the following cases:

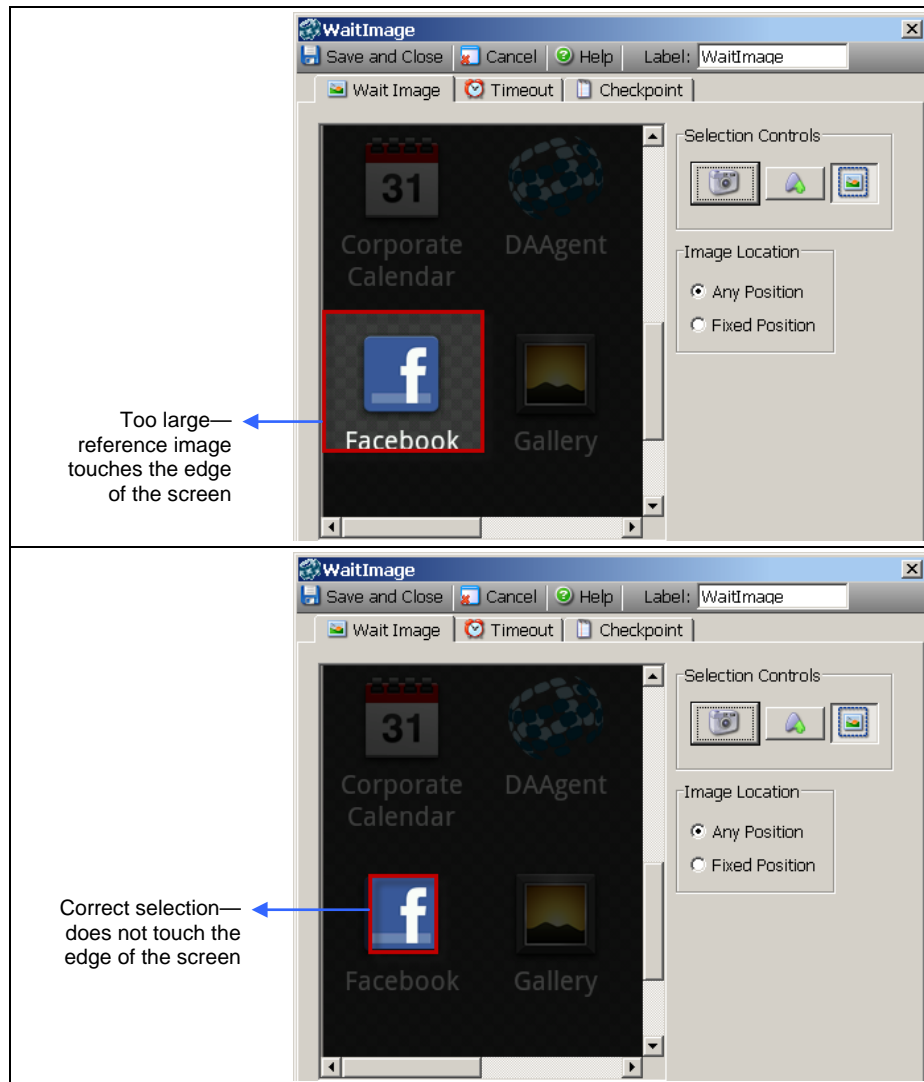
- ◆ Some device screens are darker around the edge than the center. A reference image defined on such a screen might fail if the area it references moves from the edge of the screen to the center or vice versa. (See [Looking for Android Icons](#) for an example of this scenario.)
- ◆ Some identical hardware-integrated devices might not have strictly overlapping screens. Screen edges can be slightly offset from each other on two such devices of the same make and model. A reference image defined at the screen edge of one such device might not work on another.

Figure 2-16 Screen Edges Not Perfectly Aligned



To avoid possible areas of mismatch in these cases, do not include screen edges in your reference image.

Figure 2-17 Avoid Selecting the Edge of the Device Screen

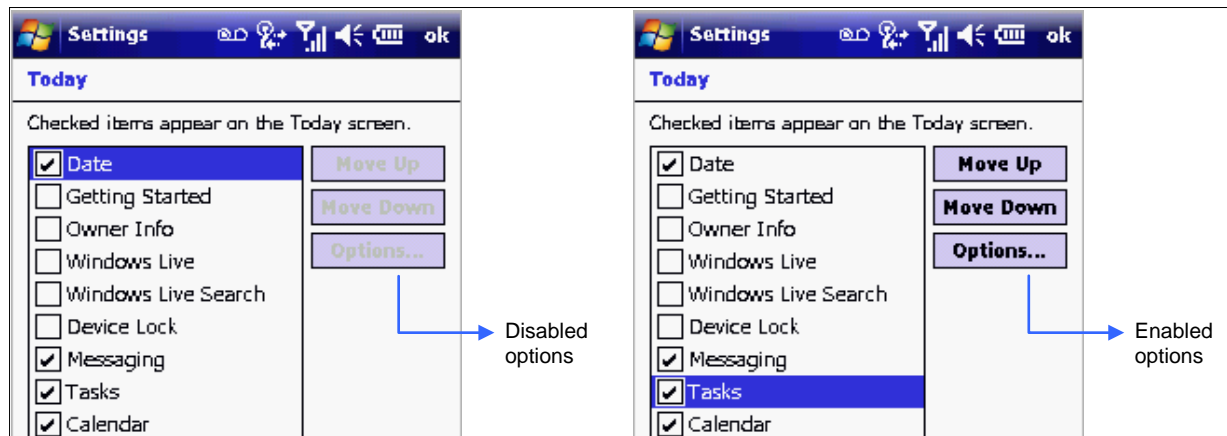


Grayed out UI Options

UI options on some device screens are grayed out until the corresponding action is valid. In the example below, buttons are enabled depending on the previous menu item selected.

When creating an image-based reference point, ensure that UI elements are in the same state in the reference image as well as the device screen or your script might fail. If a menu option is enabled in a reference image, ensure that your script enables it on the device screen as well.

Figure 2-18 Device Screen Changes Based on Disabled/Enabled Options



Selecting Application Icons

As a general rule, select an area *inside* an application icon to define a reference image. This ensures that any possible changes to the background do not affect your reference image.

Some application icons display dynamic application status information. The image below shows the Facebook icon for a BlackBerry 8900 (Curve) with no updates and then with an indicator of unread content.

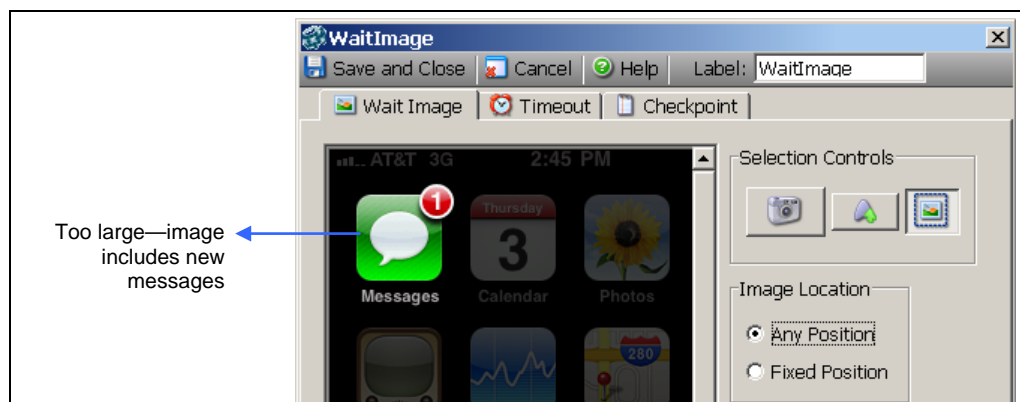
Figure 2-19 Changes to Application Icons

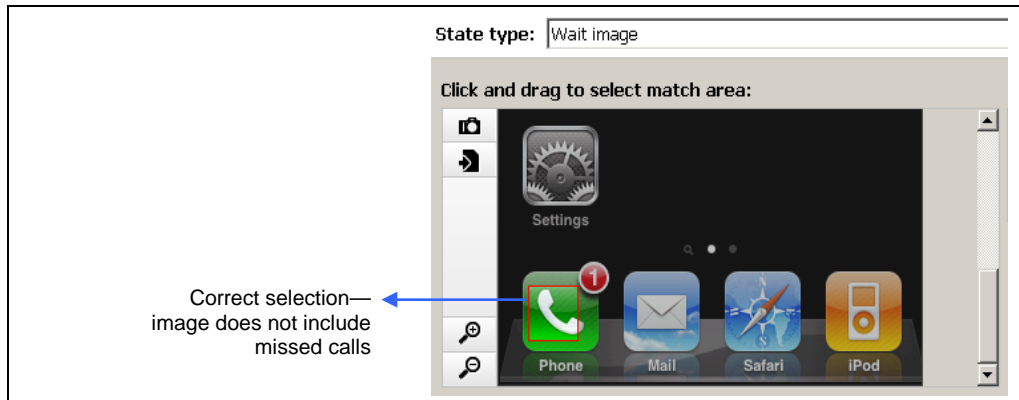


When choosing application icons as reference points, try first to use text extraction and match the application name in a string of text. If that is not possible and you need to define a reference image, be sure to exclude any part of the icon that displays application status.

For instance, if your reference image is the messaging or phone icon on an iPhone, do not include the area that displays the number of new messages or missed calls (see Figure 2-20 below). Or you can use text matching to a string containing the application name.

Figure 2-20 Choosing an Application Icon





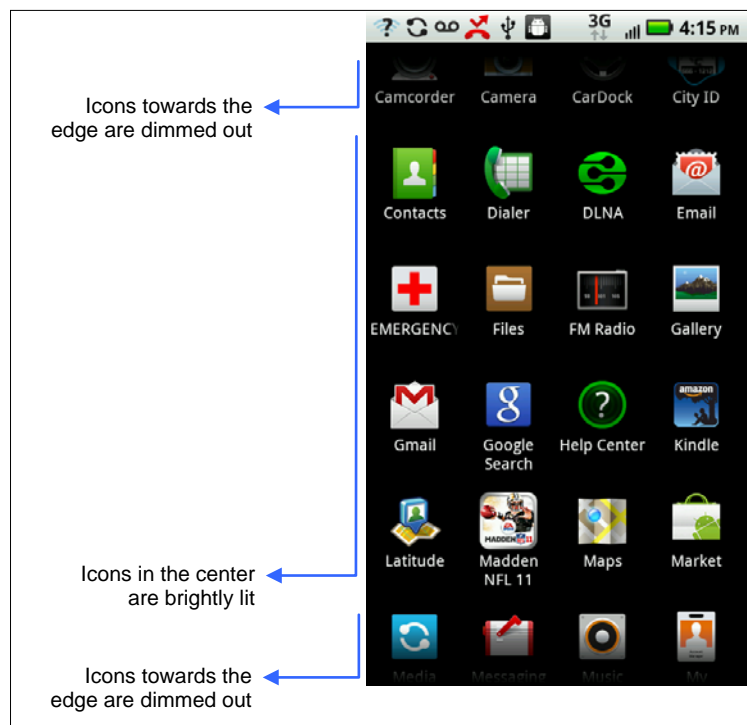
Looking for Android Icons

Figure 2-21 Android Application Icons

To simplify your scripts on Android devices, add any icons you will be using to the home screen of the device (see [Application Setup](#) above).

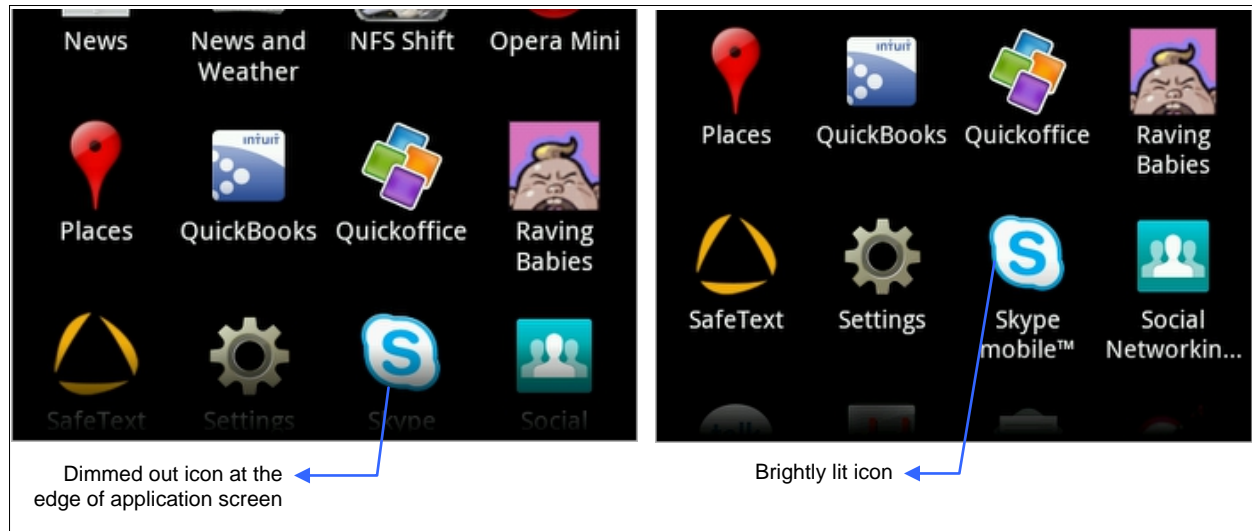
Application icons on some Android application screens are more brightly lit towards the center of the screen and dimmed out towards the edge of the screen (see Figure 2-21).

When setting an Android application icon as a reference image, move it to the center of the screen before selecting it. In your script, implement a series of swipes (using the Navigate To command) to display the same icon on the center of the screen before image matching is performed. Any attempt to match your reference image to the icon at the edge of the screen will fail because of differences in image tone.



The images below show the Skype application icon first at the edge of a Droid 2 screen, and then brightly lit in the center of the screen, illustrating how the difference in tone can cause a reference image of the icon to fail. In order to avoid match failures, select your reference image at the center of the screen. Then have the image match performed at the center of the screen.

Figure 2-22 Differences in Android Icon Tone Based on Position on the Application Screen



Searching for BlackBerry Icons on Non-Touchscreen Devices

On non-touchscreen BlackBerry devices, move the application icon you wish to work with to the first row of the icon menu if possible; this will make it easier to find and select the icon (see [Application Setup](#) above).

If you need to find and select an application icon with a dynamic location on a non-touchscreen BlackBerry device, you will need start from the top of the application screen and tab through all the icons to conduct a thorough search.

An efficient way to search would be to place a pair of Navigate To commands within a loop and set a variable as the loop condition.

Use the first Navigate To command to repeatedly tab right through a row of icons and then wait to verify an image of the browser icon. If the image is found, reset the loop. If not, navigate to the next row of icons and use the second Navigate To command to repeatedly tab left through the row and wait to find the reference image. If the image is found, reset the loop. If not, navigate to the next row of icons and execute the loop again.

The images below show a sample Navigate To command to tab right through the icons on the application screen and wait for an image of the browser icon. As the screen is five icons wide, it repeats this procedure five times or until the reference point is found, whichever comes first. In the Wait Image tab, the command uses the browser icon to define a reference image.

Figure 2-23 Navigate To Command for Tabbing Right on a BlackBerry Application Screen

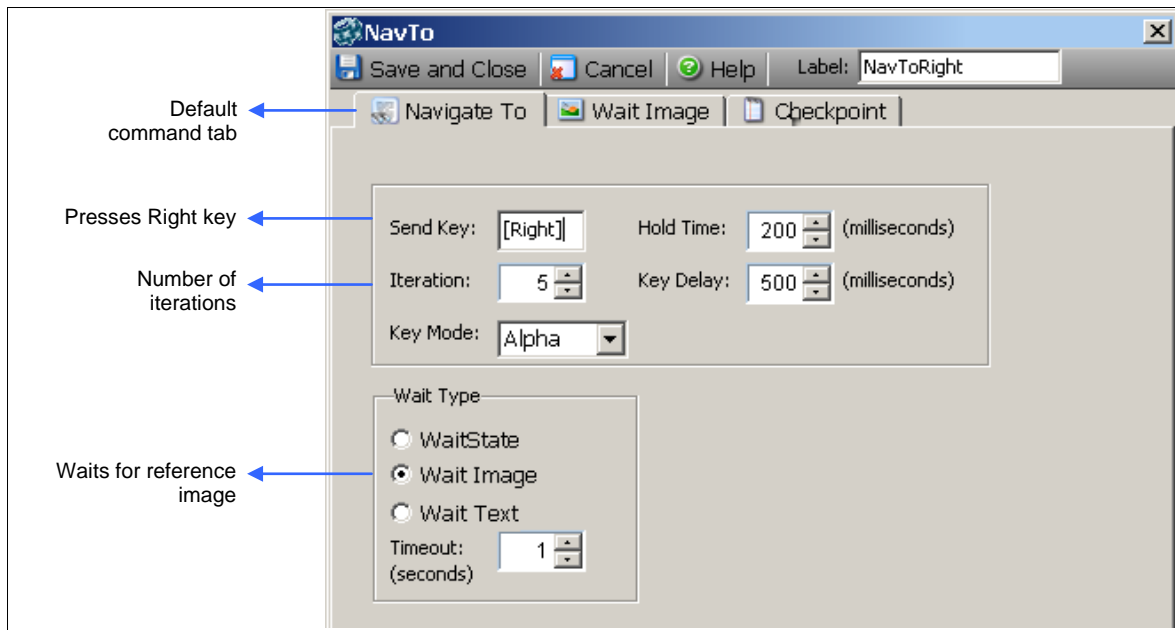
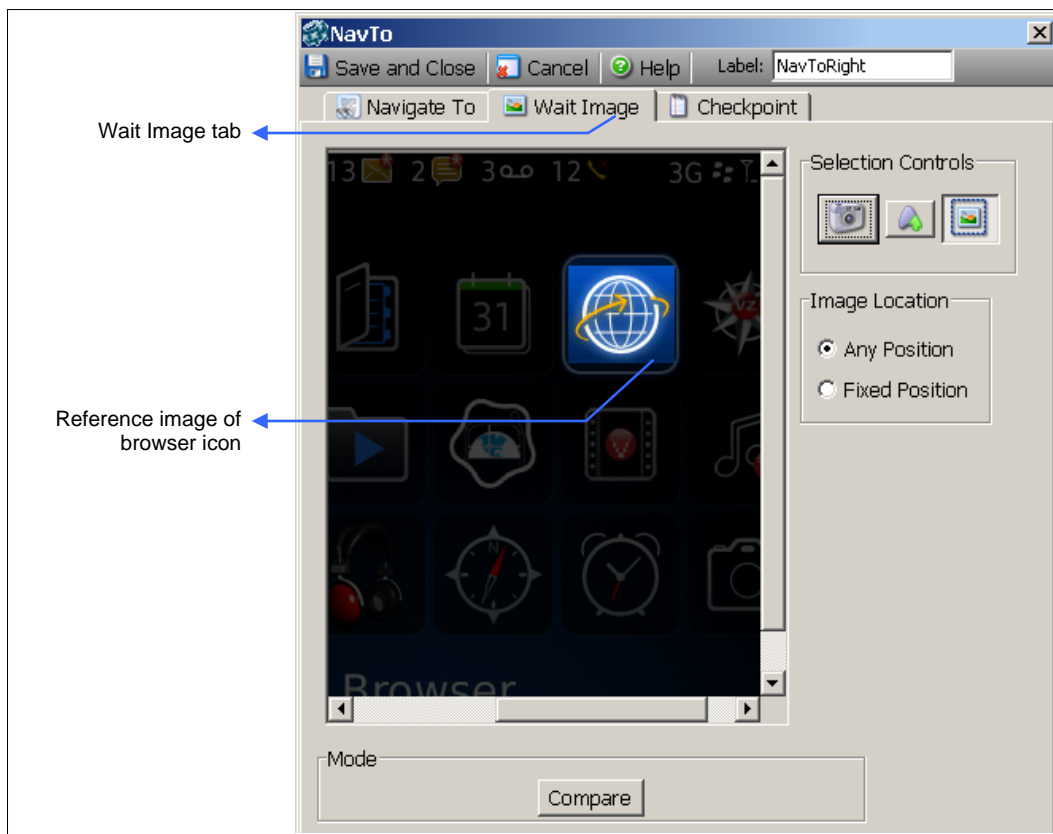
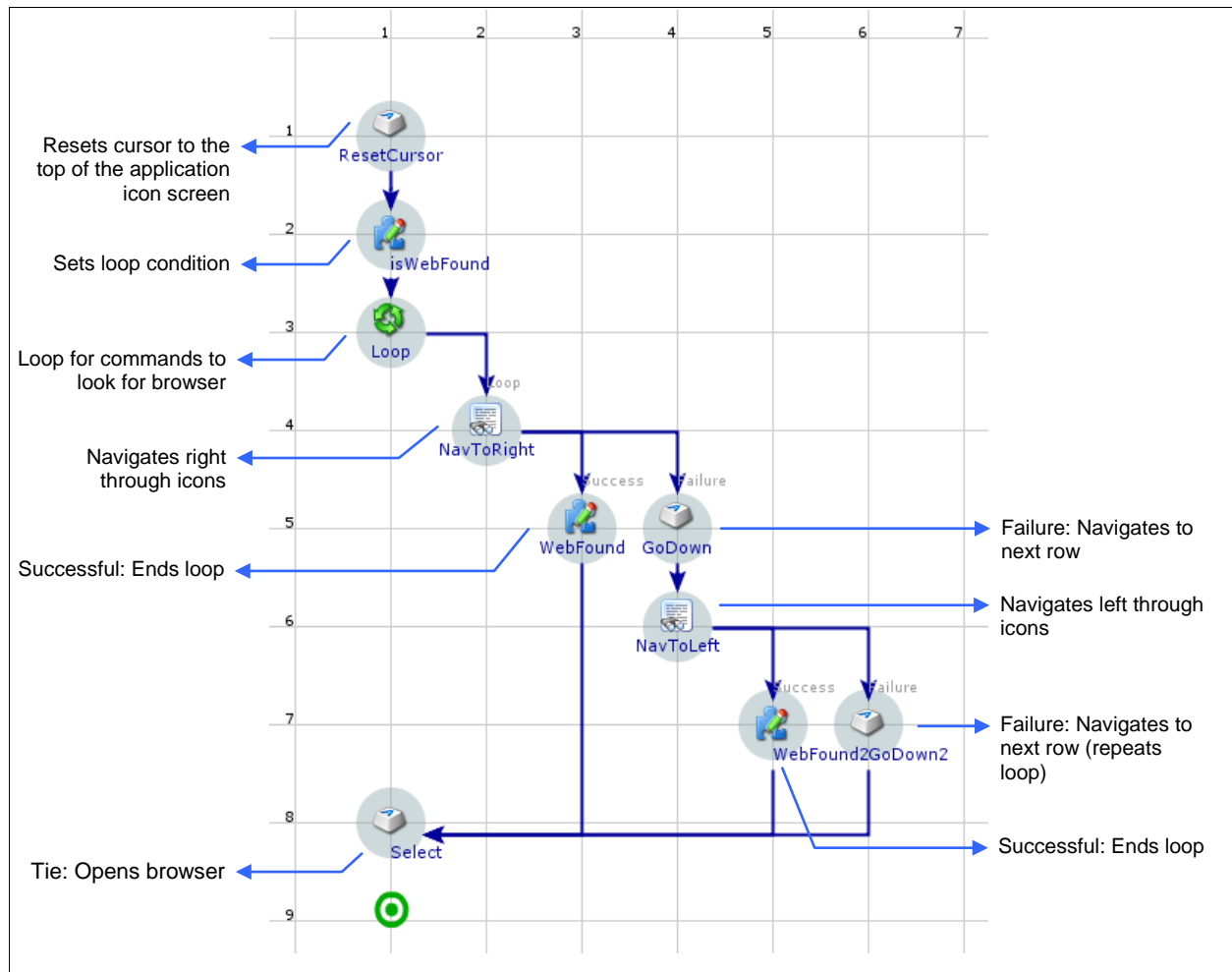


Figure 2-24 Wait Image Tab in Navigate To Command



The image below shows a script that implements this scenario to search for and select the browser icon on a BlackBerry Curve.

Figure 2-25 Using Navigate To in a Loop to Search for a BlackBerry Icon



Device Orientation

Changing device orientation (e.g., using the Hardware Extension command) does not normally impact reference images as Keynote DeviceAnywhere automatically updates the device screen to match the orientation.

NOTE Accelerometer is supported on hardware-integrated and hybrid devices. On software-integrated devices:

- ◆ Accelerometer supported for software-integrated iPhones; you can change orientation of some software-integrated Androids but video is not rotated.
- ◆ Flip open/close (with screen rotation) supported for software-integrated Androids.

However, some screens can only be viewed in the default orientation. Consequently, any reference images captured in the default orientation will fail if you rotate the device as the screen is not updated.

For example, the iPhone home or settings screen can only be viewed correctly when the device is vertical. If you flip the device on its side, these screens are not updated (see image below). Any reference images captured while the device is in portrait orientation will fail if the orientation is changed for a script run.

Figure 2-26 Some Screens Do Not Change Orientation



3 Guidelines for Advanced Settings

This chapter discusses how to use the advanced settings in commands to troubleshoot false match failures.

You would detect a false match failure if your script failed at a reference point even though images in test results do not indicate any variation from the reference image. Small changes to the device screen produce changes that are not visible to the human eye but cause match failures all the same. For example:

- ◆ After several script runs, a screen that used to match now fails.
- ◆ Your implementation covers two devices of the same model, and a reference point matches on one device but not on the other.

NOTE You would detect a *false positive match* if your script failed somewhere after the false match. You might also notice from images in test results that a match does not take place on the reference screen. A false positive match occurs either when tolerances are set too low or there are multiple match possibilities on the screen.

In advanced settings, you can:

- ◆ Compare the actual device screen to the reference image to highlight any areas of mismatch.
- ◆ Define additional match or mismatch possibilities, e.g., from test results, drag in the image of a false match failure (a device screen that should have matched but did not).
- ◆ Optimize tolerance levels for variation between the actual device screen and your reference image(s).

Additional Match and Mismatch Possibilities

While it is not necessary to specify match and mismatch image ranges, this ability is useful when your reference image seems to fail incorrectly due to small changes in device resolution or the device screen that are not visible to the human eye—you can define a range of images and adjust your tolerance settings so that any of them matches the live device screen.

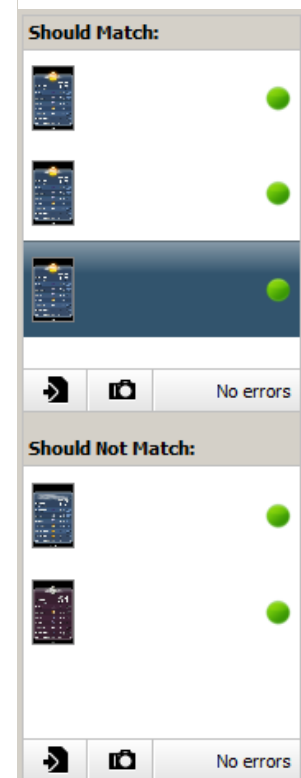
You would detect a false match failure when images in test results for a failed match do not indicate any visible difference from the base reference image. You can import the image of the device screen that was wrongly failed as an additional match possibility.

CAUTION Import only images from Keynote DeviceAnywhere test results that were directly captured from the device and uploaded to the Portal. Images from any other source might not have the resolution required for Keynote DeviceAnywhere’s pixel-to-pixel image matching technology, resulting in verification failures.

Additionally, you can define additional match/mismatch ranges:

- ◆ To delimit what can be considered a match by specifying mismatch image(s)—you can then adjust tolerance settings to ensure that the “mismatch” images always fail.

Figure 3-1 Image Library









- ◆ When the device screen you have used for a reference image has dynamic elements, and you want to specify a range of match and mismatch possibilities—you can then adjust tolerance settings to include matches and exclude mismatches.

The purpose of setting match and mismatch ranges is to arrive at a tolerance setting that is flexible enough to accommodate minor differences in the target image.

NOTE When defining a range of images to match, do not use widely different images or you will end up setting tolerance levels that produce false positive matches. If a given device interaction has two entirely different outcomes, e.g., toggling between the search functionality and the home screen by pressing the Home button on an iPhone, use Wait Event to define two distinct and entirely different reference points.

Fixing Mismatches

As you add match and mismatch candidates to the image library in the Advanced Wait Image Settings window, green  and red  icons indicate match status of each image. The table below describes what they indicate in the **Should Match** and **Should Not Match** libraries:

Library	Icon Color and Description
Should Match	 Image (match candidate) matches base reference image.
	 Image does not match base reference image.
Should Not Match	 Image (mismatch candidate) <i>correctly does not match</i> base reference image.
	 Image <i>incorrectly matches</i> base reference image.

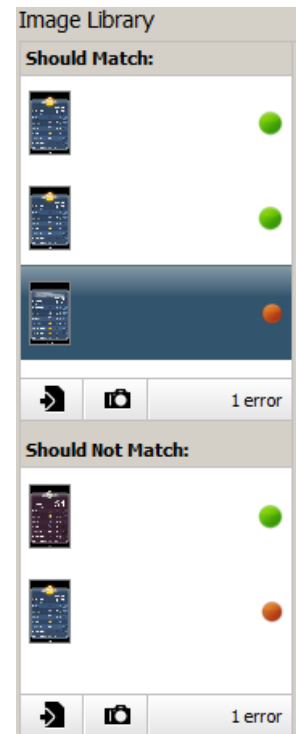
You can take the following steps, in the order listed, to ensure that all images in the image library display a green icon:

- 1 Check that you are comparing the base reference image to the correct live device screen or correct imported image, especially if mismatches (pink areas) are generalized. If you find that you have indeed added an incorrect image to the library, right-click and select **Remove From Library** to delete it.
- 2 Check whether you have made any changes to device settings that are causing the mismatch. Undo the changes if you can.
- 3 If you cannot undo changes on the device, redefine the base reference image so additional match/mismatch possibilities in the library show green icons.
- 4 Only if the measures above do not resolve the red icons, adjust [tolerance sliders](#).

Tolerance Sliders

Tolerance levels specify the required match percentage between the reference image and the device screen. Pixel tolerance specifies the percentage of pixels to be matched. Color tolerance sets the degree of color fidelity required. Position tolerance specifies the radius around each pixel’s original position in which it must be found (with 100 being the strictest match or the tightest radius).

Figure 3-2 Image Library Requiring Troubleshooting



Default tolerance levels are:

- ◆ **Pixel tolerance** – 100
- ◆ **Color tolerance** – 70
- ◆ **Position tolerance** – 100

When you adjust tolerance sliders, settings apply to all the images in the library so you can optimize tolerance levels for variations between the reference image(s) and the actual device.

When [troubleshooting red icons in the image library](#), leave tolerance sliders at default values; adjust sliders only when other measures such as checking device settings and redefining the base reference image do not work. Use the following guidelines to adjust tolerance sliders:

- ◆ Increase sliders only in small increments (e.g., up to the next notch on the scale).
- ◆ Setting tolerance levels too high causes false positive matches.