Table of Contents

[Goals 2](#_Toc191980786)

[Processes 2](#_Toc191980787)

[Setup a machine to run automation 2](#_Toc191980788)

[Run automation 2](#_Toc191980789)

[Setup a machine to code and build automation 3](#_Toc191980790)

[Building automation 3](#_Toc191980791)

[Coding automation 3](#_Toc191980792)

[Making the WTT job 3](#_Toc191980793)

[Tools 3](#_Toc191980794)

[WTT 3](#_Toc191980795)

[RPF 4](#_Toc191980796)

[Mash 5](#_Toc191980797)

[ASI 5](#_Toc191980798)

[Source Depot 5](#_Toc191980799)

[Product Studio 5](#_Toc191980800)

[Test specific notes: 5](#_Toc191980801)

[Advanced Photo 5](#_Toc191980802)

[Backup 5](#_Toc191980803)

[Bitlocker 5](#_Toc191980804)

[Branding 6](#_Toc191980805)

[CPU Sockets 6](#_Toc191980806)

[Domain Join 6](#_Toc191980807)

[DWM 6](#_Toc191980808)

[Fax and Scan 6](#_Toc191980809)

[FUS 6](#_Toc191980810)

[Games 7](#_Toc191980811)

[Group Policy 7](#_Toc191980812)

[HD Publish 7](#_Toc191980813)

[MCE 7](#_Toc191980814)

[Mobility Center 7](#_Toc191980815)

[MUI 7](#_Toc191980816)

[Network Projector 8](#_Toc191980817)

[Parental Controls 8](#_Toc191980818)

[RAM 8](#_Toc191980819)

[Remote Desktop 8](#_Toc191980820)

[SideShow 8](#_Toc191980821)

[TimeWarp 8](#_Toc191980822)

[Welcome Center 9](#_Toc191980823)

[WMP and MM 9](#_Toc191980824)

# Goals

The goal for SKUDiff Automation is to create a fast-running suite of tests that verifies SKU composition as seen through the User Interface. This automation needs to run on multiple platforms (x86 and x64), and multiple languages.

# Processes

## Setup a machine to run automation

Tools: WTT, ASI

See “SKUDiff Automation lab setup.docx”, “SKUDiff Executing Automation” and “SKUDiff Automation Requirements.docx”

## Run automation

Tools: WTT, MASH

See “Executing the weekly automation run.docx”

## Setup a machine to code and build automation

Tools: Source Depot, Razzle

Enlistment/deployment of SKUDiff package docs needed.

## Building automation

Tools: Razzle

See “SKUDiff Automation Coding.docx”

## Coding automation

Tools: WTT, Source Depot, RPF Recorder

See “SKUDiff Using RPF.docx”, “SKUDiff Automation Coding.docx”, “SKUDiff Shared Library.docx”, and “SKUDiff Building Automation”

### Making the WTT job

Tools: WTT

Copy an existing job

Search-and-replace WTT down right now, can’t fully document

# Tools

## WTT

WTT is a collection of multiple tools used across Windows testing, the key components we depend on are:

WTT Studio, the main user interface for tested to use

WTT Client, the service that runs on test machines, to execute tests

WTT Datastore, the database which tracks Machines, Jobs, Results, etc.

We also use other WTT components, such as ASI; but it’s not a requirement.

We currently install WTT from [\\TKBGITWTT61](file:///\\TKBGITWTT61), but this is subject to change.

Jobs in WTT are collections of commands to execute, separate from the test code which is executed.

Assets in WTT represent test machines, Machines are in ‘Pools’ which determine who can run what tests on them.

Results are generated by executing Jobs on machines, or on Pools with Constraints.

More information on WTT is available at <http://team/sites/WTT/help.aspx>

## RPF

RPF, “Record and Playback Framework” is a wrapper around different UI Automation modules, the key ways we interact with RPF are:

Recorder, which can log user actions and automatically generate code to repeat those actions on multiple languages. Recording in generally performed on Pseudo-Localized builds, as they guarantee that each string is unique.

Playback, a collection of functions that take QueryIDs and ResKeys to perform actions.

The automatically generated code is unfortunately not always SKU and Platform portable, and does not log results in the way we need, so we generally cut and paste bits of the generated code into our code.

There are two types of character string constants that RPF recorder generates, and often need to be manually adjusted:

QueryID, which is a chain of attributes of UI elements starting from the Desktop, used to target a specific element. The challenge of a well formed QueryID is to write it so that it always finds the element you want. More information on QueryIDs is available at [\\rpfbuilds\Release\Generation1\Version117(RC1)\Documentation\QueryId 11.doc](file:///\\rpfbuilds\Release\Generation1\Version117(RC1)\Documentation\QueryId%252011.doc)

ResKeys, which specify how and where to pull a localized string from, primarally for use by QueryIDs. Some custom resource extractors have been written for specific test cases.

We are also stuck with an older version of RPF 11.7 for Vista SP1, so be careful when relying on documentation that may not apply.

<http://rpfforum/Forums/default.aspx>

## 

## Mash

Mash is the ‘Test Harness’ we have used, it calls functions in test .DLL’s as specified by Suite (.swt) files, and provides Logging support.

## 

## ASI

Often used to install builds for testing, part of the WTT suite of tools, it is used in the current process to deploy builds for testing.

## 

## Source Depot

Source control system used for Windows, currently we store our code in Cresent\_QFE, ideally it will end up in the Shelltest area of the main Windows source.

## 

## Product Studio

The main tool used to communicate issues in software under development. Used to file bugs, research known issues, find area owners, etc.