# Quick Reference Book Auto-IT

java.lang.Runtime.*getRuntime*().exec("C:/Program Files/AutoIt/Authenticate.exe");

message box:

Syntax: MsgBox(Flag, "Title", "Message")… Flag Value- changes the way MsgBox. EX: 64

MsgBox(0, "Tutorial", "Hello World!")

message flag

0 –OK

1 –OK, Cancel

2 –Abort, Retry, Ignore

3 –Yes, No, Cancel

4- Yes, No

5 –Retry, Cancel

6 –Cancel, Try Again, Continue

---------------------------------------------------------------------------------------------

Notepad

Run("notepad.exe")

Syntax(wait):WinWaitActive ( "title" [, "text" [, timeout = 0]] )

WinWaitActive("Untitled - Notepad")

Send("This is some text.")

WinClose("Untitled - Notepad")

Syntax :WinExists ( "title" [, "text"] )

if(WinExists ( "title" [, "text"] )){…DoSomething}

text- check for text in window.

;check Save text in Notepad popUP during closing.

WinWaitActive("Notepad", "Save")

ALT-N:close popUp

Send("!n")

---------------------------------------------------------------------------------------------

Some Window with Setup button

WinWaitActive("WinZip® 9.0 SR-1 Setup", "&Setup")

ALT S:setup

Send("!s")

Send("{ENTER}")

ALT Y:yes

Send("!y")

---------------------------------------------------------------------------------------------

RegEx matching:

StringRegExp( "testString" , "pattern" [, flag ] )

MsgBox(0, "SRE Example 1 Result", StringRegExp("text", 'test'))

;output-0

MsgBox(0, "SRE Example 2 Result", StringRegExp("text", 'te[sx]t'))

;output-1

MsgBox(0, "SRE Example 3 Result",StringRegExp("text", 't{1}e{1}[sx]{1}t{1}'))

EX:2 flag 1, returns matched String.

$asResult = StringRegExp("This is a test example", '(test)', 1)  
If @error == 0 Then  
    MsgBox(0, "SRE Example 4 Result", $asResult[0])  
EndIf  
$asResult = StringRegExp("This is a test example", '(te)(st)', 1)  
If @error == 0 Then  
    MsgBox(0, "SRE Example 4 Result", $asResult[0] & "," & $asResult[1])  
EndIf

---------------------------------------------------------------------------------------------

**GUI dialog**

#include <GUIConstantsEx.au3>  
#include <StaticConstants.au3>  
#include <EditConstants.au3>  
#include <WindowsConstants.au3>

**;Title** = "Calculator" **Width** = 260 **Height** = 230

GUICreate("Calculator", 260, 230)

**;Text** ="1", **Left** = 54, **Top** = 138, **Width** = 36, **Height** = 29

$CTRL\_btn1 = GUICtrlCreateButton("1", 54, 138, 36, 29)

Local $CTRL\_btn0 = GUICtrlCreateButton("0", 54, 171, 36, 29)

Edit screen

Local $CTRL\_EdtScreen = GUICtrlCreateEdit("0.", 8, 2, 239, 23)

Local $CTRL\_LblMemory = GUICtrlCreateLabel("", 12, 39, 27, 26)

Show GUI

GUISetState()

edit screen styling

Local $CTRL\_EdtScreen = GUICtrlCreateEdit("0.", 8, 2, 239, 23, BitOR($ES\_READONLY, $ES\_RIGHT), $WS\_EX\_STATICEDGE)

Local $CTRL\_LblMemory = GUICtrlCreateLabel("", 12, 39, 27, 26, $SS\_SUNKEN)

---------------------------------------------------------------------------------------------

Create maven project:

<http://maven.apache.org/guides/getting-started/maven-in-five-minutes.html>

step1: create project and open cmd window there and run the following:

mvn archetype:generate -DgroupId=packageName -DartifactId=my-app -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

step2: move command to newly created my-app directory

cd my-app

step3: run package (build jar)

mvn package

step4: Execute the jar

java -cp target/my-app-1.0-SNAPSHOT.jar packageName.App

OR

Download JDK 1.X.X (Some version)

Download Eclipse

Download Maven (Configure path)

Download M2Eclipse plugin

Create new maven Project in Eclipse.

---------------------------------------------------------------------------------------------

Ant Project:

<http://www.vogella.com/tutorials/ApacheAnt/article.html>

1. Create package java in src

2. Create package test in src

3. Add file to src and name it build.xml

4. Add these code to build.xml

<project name="Ant-Test" default="main" basedir=".">

<!-- Sets variables which can later be used. -->

<!-- The value of a property is accessed via ${} -->

<property name="src.dir" location="src" />

<property name="build.dir" location="bin" />

<property name="dist.dir" location="dist" />

<property name="docs.dir" location="docs" />

<!-- Deletes the existing build, docs and dist directory-->

<target name="clean">

<delete dir="${build.dir}" />

<delete dir="${docs.dir}" />

<delete dir="${dist.dir}" />

</target>

<!-- Creates the build, docs and dist directory-->

<target name="makedir">

<mkdir dir="${build.dir}" />

<mkdir dir="${docs.dir}" />

<mkdir dir="${dist.dir}" />

</target>

<!-- Compiles the java code (including the usage of library for JUnit -->

<target name="compile" depends="clean, makedir">

<echo message="Compiling Code..." />

<javac includeantruntime="false" srcdir="${src.dir}" destdir="${build.dir}">

</javac>

</target>

<!-- Creates Javadoc -->

<target name="docs" depends="compile">

<echo message="Compiling Code..." />

<javadoc packagenames="src" sourcepath="${src.dir}" destdir="${docs.dir}">

<!-- Define which files / directory should get included, we include all -->

<fileset dir="${src.dir}">

<include name="\*\*" />

</fileset>

</javadoc>

</target>

<!--Creates the deployable jar file -->

<target name="jar" depends="compile">

<echo message="Creating jar..." />

<jar destfile="${dist.dir}\de.vogella.build.test.ant.jar" basedir="${build.dir}">

<manifest>

<attribute name="Main-Class" value="test.testClass" />

</manifest>

</jar>

</target>

<target name="run" depends="compile, jar, docs">

<echo message="Running Test..." />

<java jar="${dist.dir}\de.vogella.build.test.ant.jar" fork="true" />

</target>

<target name="main" depends="compile, jar, docs , run">

<description>Main target</description>

</target>

</project>

5. Run build.xml file (run as- ant build)

6. In project Directory run cmd,

ant -f build.xml

7. Run ant (default command),

ant

8 Can run single targets as,

ant compile

ant jar

ant run

-----------------------------------------------------------------------------