# Wappsign User's Guide

Version 1.0

# Important information

Texas Instruments makes no warranty, either expressed or implied, including but not limited to any implied warranties of merchantability and fitness for a particular purpose, regarding any programs or book materials and makes such materials available solely on an “as-is” basis.

In no event shall Texas Instruments be liable to anyone for special, collateral, incidental, or consequential damages in connection with or arising out of the purchase or use of these materials, and the sole and exclusive liability of Texas Instruments, regardless of the form of action, shall not exceed the purchase price of this product. Moreover, Texas Instruments shall not be liable for any claim of any kind whatsoever against the use of these materials by any other party.

The latest version of this Guide, along with all other up-to-date information for developers, is available at http://education.ti.com.

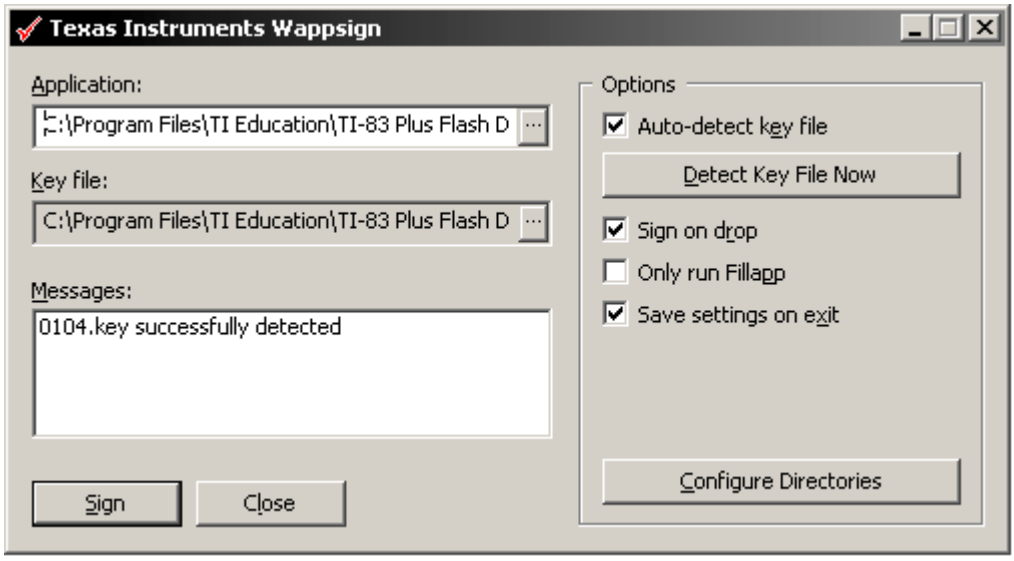
 

# 1 Introduction

Wappsign is the Windows™ equivalent of the “appsign” utilities previously provided with the TI-83 Plus SDK. Wappsign has a few additional features to make signing applications even easier as well providing scripting support so developers can automate their signing process.

## User Interface

The interface of Wappsign is very simple. When you start the application, you should see a window like the one below:




Wappsign has the ability to automatically detect what key file to use to sign an application. It does this by maintaining a list of directories (up to 10) where you keep your key files. When Wappsign starts for the first time it will add the “TI83PLUSDIR\utils” directory to its search path if the SDK is detected.

# 2 Using Wappsign

## Signing an application

With the auto-detect feature, signing in Wappsign couldn't be easier. Type the path to your hex file, or click the browse button ("…") to browse for the file. Wappsign will automatically detect the key file from the application header and, if found, display the path to the key file and the message "XXXX.key successfully detected" in the message window (XXXX = the detected key file). Click on the "Sign" button, and if there are no problems, the application will be generated in the same directory as the *.hex file with the same name. Wappsign detects the calculator the application has been written for (either TI-83 Plus or TI-73) and generates the *.8xk or *.73k file accordingly.

Wappsign supports drag and drop to make signing applications even easier. Select the "Sign on drop" option to enable this feature. Drag the *.hex file onto the Wappsign window and the *.hex file is automatically put into the "Application" and any *.key file is put in the "Key file" field. Additionally, if "Auto-detect key file" is selected, the "Sign on Drop" option is available. This will sign a dropped application if a key file is successfully detected making application signing as simple and "drag'n'drop".

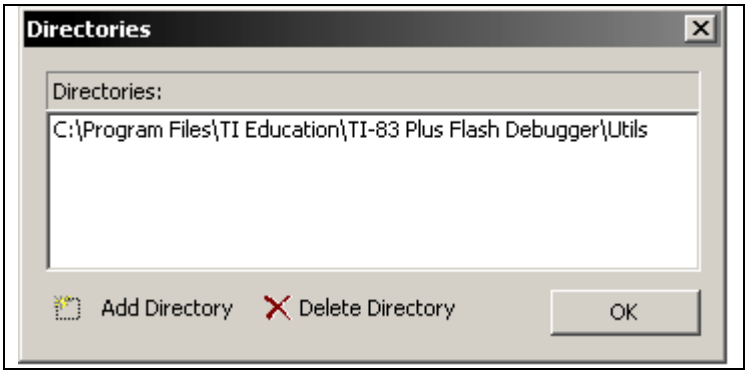
## Only Running Fillapp

When developing multpage applications, the *.hex file must be "filled" before attempting to run the app in the TI-83 Plus Simulator/Debugger. Wappsign gives you the option to only fill the application, and not attempt to sign it. Simply select the "Only Run Fillapp" option. The "Sign" button will now say "Fill", and clicking on "Fill" will fill the application. The original *.hex file will be overwritten by the filled *.hex file.

# Managing Directories

Wappsign keeps a list of up to ten directories in which the application will search for key files. You can edit this list of directories by clicking the "Configure Directories" button. When you click on Configure Directories, you should see a window similar to the one below:




New directories can be added by clicking the "Add Directory" button. You can delete a directory from the list by highlighting the directory and clicking "Delete Directory".

Alternatively, you can manually select a key file at any time by clicking the browse button ("…") for the key file field. Find and select the key you want to use. If the directory is not in your search path, Wappsign will prompt you to add it.

# Errors

A few things can go wrong when signing but Wappsign will try to help you. Most of the errors come with small suggestions to remedy the problem. To see a more verbose listing of Wappsign's activities, you can look at the log file Wappsign generates each time it is run. The log file will be generated in the same directory as the input file.

# 3 Developer's Information

## Introduction

Wappsign is designed to aid developers make the signing process as easy as possible. To this end, Wappsign will expose its signing interface through an automation object. This enables a developer to create applications that sign hex files in any COM-enabled language or technology – Microsoft™ Visual Basic, Microsoft™ Visual C++, Windows Scripting Host or ASP to name a few. This section will briefly outline the Wappsign interface and give an example.

## Properties

*long Flags*

Flags can be any one or combination of the following:

| | |
|---|---|
| waVerbose = 1 | - Generates the log file. |
| waOutput83 = 2 | - Affects FormatOutput(). Appends .8xk. |
| waOutput73 = 4 | - Affects FormatOutput(). Appends .73k. |
| waFillOnly= 8 | - Will append .hex to the output file and only run the Fillapp. |
| waDetectType = 4096 | - Detects the target calculator type (83P/73). |

## Methods

*long Sign(String szHex, String szKey, String szOutput)*

Signs *szHex* using *szKey* to produce *szOutput*.

Return Value: 0 if signed correctly, otherwise an error code. Use GetErrorMessage() to retrieve the full text error message. If waDetectType is used (see above), then Wappsign detects target calculator type.

*String GetKeyFile(String szHex)*

Detects and retrieves the key file used by the application. Uses the search paths saved in the registry. These can be modified at run time using the directory functions listed below.

Return Value: A string with the full path name to the key file, otherwise "".

*String FormatOutput(String szFilename)*

Simply takes szFilename, removes the extension and adds either .8xk, 73k or .hex according to the Flags properties (see above).

Return Value: Output filename, otherwise "".

*String GetErrorMessage(long lError)*

Returns the full error message given an error code.

Return Value: Will return the error message. Passing 0 returns "WAPP0000: Successful sign!" and passing an invalid or unrecognized error will return "XXXX0000: An unspecificed error has occurred!"

*Boolean AddDirectory(String szDirectory)*

Adds a directory to the search path. All trailing slashes are removed.

Return Value: Returns true if the directory was successfully added. False if 10 directories are already in the search path.

*Boolean RemoveDirectory(String szDirectory)*

Removes a directory from the search path.

Return Value: True if directory was successfully removed. False if directory could not be found or removed.

*String GetDirectory(int iIndex)*

Retrieves the iIndex-th directory from the search path.

Return Value: Directory path name. Returns "" if the index is out of range.

# Example

This Windows Script will sign a hex file that is passed to it. The hex file is signed by running Windows Scripting Host with the arguments or by dragging and dropping a hex file on to the script icon. The script is designed to always produce output – if the keyfile cannot be found, it will fill the hex file instead.

```
<job id="SignHex">

<SCRIPT Language="VBScript" src="WappConsts.vbs"/>

<SCRIPT Language="VBScript">
  Option Explicit

  Dim sHexFile
  Dim sKeyFile
  Dim sOutput
  Dim oWappsign
  Dim lErrorCode
  Dim objArgs

  Set objArgs   = WScript.Arguments
  Set oWappsign = CreateObject("Wappsign.Sign")

  If objArgs.Count = 0 Then
    MsgBox "Drag a hex file on to this script icon to sign
it"
    WScript.Quit
  End If

  oWappsign.Flags = waVerbose

  sHexFile  = objArgs(0)
  sKeyFile  = oWappsign.GetKeyFile(sHexFile)
```

```
  If sKeyFile = "" Then
    MsgBox "Key file could not be found - Fillapp only
build."
    oWappSign.Flags = waVerbose Or waFillOnly
  End If

  sOutput   = oWappsign.FormatOutput(sHexFile)

  lErrorCode = oWappsign.Sign(sHexFile, sKeyFile, sOutput)

  If lErrorCode <> 0 Then
    MsgBox oWappsign.GetErrorMessage(lErrorCode)
  End If

</SCRIPT>

</job>
```

The code is quite self-explanatory. The job and script tags are merely used to add tell the Windows Script Host a little more about the script itself. "WappConst.vbs" is a VBScript file that includes all Wappsign constants:

```
' Wappsign Flags include

Const waVerbose = 1
Const waOutput83 = 2
Const waOutput73 = 4
Const waFillOnly = 8
Const waDetectType = 4096
```

Texas Instruments U.S.A.
7800 Banner Dr.
Dallas, TX  75251



ti-cares@ti.com
education.ti.com