New Features PocketBuilder™ 2.5

Document ID: DC00765-01-0250-02

Last revised: February 27, 2008

Vista support

PocketBuilder™ 2.5 runs on the Microsoft Vista platform, although to develop and deploy PocketBuilder applications, you might need to run PocketBuilder as an administrator. To start PocketBuilder as an administrator, you can select Programs>Sybase>PocketBuilder 2.5 from the Windows Start menu, then right-click PocketBuilder in the cascaded menu, and select "Run as administrator" from the pop-up menu.

Since the Program Files directory on Vista has special properties, you should copy the PocketBuilder Code Examples directory to a different location before you deploy the Code Example applications.

Vista does not ship with the Windows standard help display program *WinHlp32.exe*. If you want to display the PocketBuilder Help, you can download this file from links on the Microsoft Web site at http://go.microsoft.com/fwlink/?LinkID=82148.

Initialization file location

To facilitate PocketBuilder support on Windows Vista, the PocketBuilder initialization file, *PK.INI*, is installed in the current user's private application data directory. For Windows XP and Windows 2003, this is typically the *C:\Documents and Settings\userName\Local Settings\Application Data* directory, where *userName* is the name of the current PocketBuilder user. For Windows Vista, this is typically the *C:\Users\userName\AppData\Local* directory.

A copy of the *PK.INI* file is installed in the same directory as the PocketBuilder executable file. PocketBuilder no longer uses the initialization file in this directory, however, you can still change the location of the operative *PK.INI* file in the System Options dialog box. You might want to do this if you use more than one version of PocketBuilder or if you are running PocketBuilder over a network.

For information on changing the location of the operative *PK.INI* file, see the chapter on customizing PocketBuilder in the *Users Guide*.

SQL Anywhere 10.0.1

The PocketBuilder 2.5 setup program includes an option for installing SQL Anywhere® 10.0.1. SQL Anywhere 10.0.1 includes the SQL Anywhere (formerly Adaptive Server™ Anywhere or ASA) and UltraLite® relational database systems, as well as the MobiLink™ synchronization server.

You can use either version 9.x or 10.0.1 SQL Anywhere components with PocketBuilder 2.5. Changes to PocketBuilder allow you to select and use the latest versions of these components, although SQL Anywhere 10 is now the default selection where a choice is required. For example, although you do not need to add the `driver=dbodbc10.dll` parameter to a DSN file specifying a SQL Anywhere 10 database, you must include `driver=dbodbc9.dll` in a DSN file specifying an ASA 9 database.

About the ASADemo_10 database

The PocketBuilder setup program installs the ASADemo_10 database in the *Code Examples\SADemoData\SA10* directory. This database is a migrated version of the Adaptive Server Anywhere 9 Sample database that you can use with SQL Anywhere 10. (It is slightly different from the demo database that installs with SQL Anywhere 10.) You must create an ODBC data source name for this database in the ODBC Administrator before you can connect to it at design time.

Upgrade to SQL Anywhere 10.0.1
The ASADemo_10 database was created with the SQL Anywhere 10.0.1 version that installs with PowerBuilder®. It cannot be used with SQL Anywhere 10.0.0. Although you can connect to SQL Anywhere 10.0.0 databases with SQL Anywhere 10.0.1, the reverse is not true. You must upgrade to SQL Anywhere 10.0.1 to use the ASADemo_10 database.

Using UltraLite 10

If you use UltraLite 10 in the design time environment, you must copy UltraLite DLLs from the SQL Anywhere *Ultralite\Win32\386\lib\vs7* directory to the main PocketBuilder directory. In PocketBuilder 2.5, the UltraLite Synchronization wizard has been upgraded to work with UltraLite 10 databases, as well as with UltraLite 9 databases.

Source control enhancements

Source control enhancements include the following:

- The Library painter shows the SCC status in a column that can be sorted
- The System Tree no longer opens or closes randomly for an SCC workspace
- Performance improvements to the GetLatestVersion functionality

  To improve performance, you can prevent objects in the workspace from being refreshed within a specified time of an earlier refresh or GetLatestVersion call. You do this by adding the "sccpbgrefresh" setting to the [Library] section of the *PK.INI* file and assigning a value to indicate the time period in minutes that must elapse before a new refresh call is made. You can modify this setting independently of the workspace's RefreshRate property.

  If you use Serena PVCS source control systems, you can add the "sccmaxarraysize" setting to the [Library] section of the *PK.INI* file to cut down on the number of API calls to the source control server. You can experiment with the "sccpbgrefresh" and "sccmaxarraysize" settings to determine the optimum values for your environment. A typical starting point for optimization would be:

  ```
  [Library]
  sccpbgrefresh=20
  sccmaxarraysize=100
  ```

  For these settings, the cached PKG files are considered valid for 20 minutes, and the maximum number of API calls for each invocation of GetLatestVerssion is 100.
- You can prevent the display of a message box when you open an object that is not checked out

You can clear the “Show warning when opening objects not checked out” check box on the Source Control tab of the Workspace Properties dialog box to prevent warning messages from displaying when an object is not checked out to you. If you do not manually clear this check box, you can still cause it to be cleared automatically by selecting the “Do not show this message again” check box on any warning dialog box that displays when you try to open an object under source control that is not checked out.

If you want to re-establish the warning messages, select the “Show warning when opening objects not checked out” check box on the Source Control tab of the Workspace Properties dialog box.

Additional uses of the SetRuntimeProperty function

The SetRuntimeProperty function lets you set global properties and modify them at runtime.

EnableExtraActivates property

In PocketBuilder 2.5, you can use the SetRuntimeProperty system function to set the EnableExtraActivates property and increase the number of window activation and deactivation messages you receive from an application running on a mobile device.

Windows operating systems typically provide fewer window activation and deactivation messages for mobile devices than for desktop computers. This is especially true for pop-up windows. Setting the EnableExtraActivates property to true enables you to capture more window activation and deactivation messages on a mobile device without breaking legacy behavior. Unfortunately, it can also cause a window to receive two identical activation or deactivation messages. However, you can turn this property on or off at any time to fine tune the message capturing behavior.

You can use the following script to capture additional windows activation and deactivation messages on a mobile device:

```
integer li_return
li_return = SetRuntimeProperty &
    + (“EnableExtraActivates”, true)
```

MOPEnable property

In PocketBuilder 2.5, you can disable MOP views, or allow the end user to disable MOP views by setting the MOPEnable property to false. This can be convenient if you want MOP views turned off only for a series of dialog boxes, after which, you can enable MOP views again by setting the MOPEnable property to true. It can also be useful if a MOP view results in an undesired layout, permitting the end user to regain control over the screen display.

When you turn off the MOPEnable property, the last values saved for X, Y, Width, and Height properties determine the positions and dimensions of windows and controls regardless of the screen orientation of the handheld device.

MOP views are enabled by default. You can use the following script to disable MOP views on a particular device:

```
integer li_return
li_return = SetRuntimeProperty &
    + ("MOPEnable", false)
```

For more information on the SetRuntimeProperty function and the other properties that it allows you to modify, see the *PowerScript Reference* in the online Help.

Support for the CooTek TouchPal keyboard

In addition to the stock software keyboards and keypads from Microsoft, and the Fitaly software input keyboard from Textware Solutions, PocketBuilder 2.5 includes support for the CooTek TouchPal pop-up keyboard. To use the TouchPal keyboard with PocketBuilder applications, you must assign `SIPTouchPal!` as the input method argument in a SetSIPType call.

For more information, see SetSIPType in the online Help.

Support for the Power event and the NTag property

PocketBuilder 2.5 adds a system Power event that occurs on notification of power state changes on a mobile device. This event is not supported on Pocket PC 2002 devices. Other devices might support this event only partially, with the level of support depending on the specific device model.

The NTag property is available on all visible controls. It allows you to assign a numeric tag value that you want to associate with the control. You can assign a numeric value to the control in a script or you can select a value on the General tab of the control's Properties view. The NTag property can be useful for voice enablement functionality, but it is up to you how you use this field.

For more information about the Power event and the NTag property, see the online Help.

Support for device power supply properties

The Environment object includes eight new properties for determining the status of the device battery or power source. The following table lists and describes these properties.

Environment object property	Datatype	Description
PowerBatteryCharging	Boolean	Indicates whether the battery is being charged.
PowerBatteryDataValid	Boolean	Indicates whether the values for PowerBatteryCharging, PowerBatteryFlags, PowerBatteryLifeRemaining, and PowerBatteryPercentage are valid. Values are: TRUE – Power battery values are valid FALSE – The device is unable to report the power battery values
PowerBatteryFlags	ULong	Reports the device battery charge status flags. Values and their meanings are: • **0** Unknown • **1** High charge state • **2** Low charge state • **4** Critically low charge state • **8** Currently charging (PowerBatteryCharging is true) • **128** No battery is present Combinations of the above values are possible. For example, a value of 10 would mean that the battery has a low charge state but is also currently charging.
PowerBatteryLifeRemaining	ULong	Estimated battery life remaining in seconds.
PowerBatteryPercentage	Integer	Percentage of full battery charge remaining.
PowerLineBackup	Boolean	Device is on backup power.

Environment object property	Datatype	Description
PowerLineDataValid	Boolean	Determines whether the values for PowerLineOn and PowerLineBackup are valid. Values are: TRUE – Power line values are valid FALSE – The device is unable to report the power line values
PowerLineOn	Boolean	Device is plugged in to main power source.

For more information, see GetEnvironment and the Environment object in the online Help.

Support for bit manipulation system functions

PocketBuilder 2.5 includes bit manipulation functions for numbers with the unsigned long datatype. The following table lists and provides brief descriptions of these functions.

Bit manipulation function	Allows you to
BitwiseAND	Compare the binary values of two unsigned long variables
BitwiseClearBit	Clear the bit from a specified position in a binary number
BitwiseGetBit	Check whether a bit is turned on at a specified position of a binary number.
BitwiseNOT	Get the bitwise complement of a binary number
BitwiseOR	Compare the bits from two binary numbers in an inclusive OR operation
BitwiseSetBit	Set the bit at a specified position of a binary number
BitwiseShiftLeft	Modify a binary number by shifting all its bits to the left by a specified amount
BitwiseShiftRight	Modify a binary number by shifting all its bits to the right by a specified amount
BitwiseXOR	Compare the bits from two binary numbers in an exclusive OR operation

For more information, including function syntax, see the *PowerScript Reference* in the online Help.

Hexadecimal conversion formats

PocketBuilder 2.5 allows you to use the String function to convert numeric data to hexadecimal formatting. The statements in the following table apply hexadecimal formatting to the decimal value 12:

Statement	Result
`String(12,"hex")`	C
`String(12,"hex2")`	0C
`String(12,"hex4")`	000C
`String(12,"hex8")`	0000000C

You can use the Integer or Long functions to convert a hexadecimal string to a decimal value. The following table shows return values for different hexadecimal strings:

Statement	Result
`li_foo = Integer ("0xC")`	12
`ll_foo = Long ("0x3ABC")`	15036
`ll_foo = Integer ("0x3ZZQ")//invalid hex value`	0

Additional support for file functions

PocketBuilder 2.5 adds support for the *initdir* and *aFlag* arguments of the GetFileOpenName and GetFileSaveName system functions. You can also use a string array for the *filename* argument of these functions. However, on a handheld device, the value of the *pathname* argument must still be the *My Documents* directory or one of its subdirectories.

For more information, see GetFileOpenName and GetFileSaveName in the online Help.

MOP positions saved in source code

When you save a window with a Multiple Orientation Painter view, PocketBuilder 2.5 includes view-specific positions and dimensions in the source code definitions for all visual objects and controls on the window. The source code inludes view ID numbers that PocketBuilder associates with the position and dimensions of a particular object or control. Now, when you export a window, all MOP views are saved to the exported file. In previous PocketBuilder releases, the source code included the positions and dimensions of an object or control only for the last view that you saved.

You can also select the Unconstrained view, based on custom size selections, in the PocketBuilder MOPView Manager.

For information about the view ID numbers, the MOPView Manager, and the MOP painter, see the chapter on "Working with Windows" in the *Users Guide*.

DLL repackaging

PocketBuilder 2.5 deployment uses smaller footprint DLLs, some of which are not required for all applications. For devices that do not have a large, single memory slot to accomodate the PocketBuilder virtual machine in a single DLL, multiple smaller memory slots might still be available. The smaller DLLs allow you to deploy and run PocketBuilder applications on devices with fragmented memory.

The PocketBuilder WinCE\arm directory contains a version of each DLL for the Pocket PC, and the WinCE\sparm directory contains a version of each DLL for Smartphone devices.

The following table lists the new DLL files that have been split off from the PocketBuilder VM.

DLL	Description
PKDWE25.DLL	DataWindow® engine support
PKHDW25.DLL	Hardware support for barcode, RFID, and fingerprint readers
PKPHN25.DLL	Phone support
PKPOOM25.DLL	Pocket Outlook support
PKICN25.DLL	Standard bitmaps and icons (PocketBuilder installs two different versions of this file for the Pocket PC. See Icon and bitmap resource file for details.)

The following table lists additional DLL files that you can (or must, in the case of the PKVM25.DLL) deploy with PocketBuilder.

DLL	Description
PKVM25.DLL	Virtual machine (required for all users)
PKBGR25.DLL	Graph support engine
PKODB25.DLL	ODBC database support
PKUL925.DLL	UltraLite 9 database support
PKUL1025.DLL	UltraLite 10 database support
PKSMS25.DLL	SMS receiver interceptor
PKTDY25.DLL	Today item screen support

Enhanced CAB Generation Tool

You can use the Enhanced CAB Generation tool to deploy any or all PocketBuilder support DLLs with your PocketBuilder application. If you select the Include PocketBuilder Support DLLs check box on the PocketBuilder Options tab of this tool, the tool will add the *PKDWE25.DLL* and *PKVM25.DLL* files to the CAB file it generates. This check box selection also enables check boxes and radio buttons for inclusion of each of the other DLLs.

Icon and bitmap resource file

PocketBuilder 2.5 installs four basic versions of the *PKICN25.DLL* file containing PocketBuilder icons and bitmaps. Two of the versions, one for the desktop and the other for handheld devices, contain standard images from previous releases of PocketBuilder. The other two versions, including one for the desktop and one for handheld devices, contain updated icons and bitmaps that are used in the most recent releases of PowerBuilder.

The PocketBuilder setup program installs the *PKICN25.DLL* files containing the older images in PocketBuilder *Support\oldicons* subdirectories labeled *device*, for the handheld images, and *desktop*, for the desktop images. It installs the *PKICN25.DLL* files containing the newer images in *Support\newicons* subdirectories that are also labeled *device* and *desktop*.

Copies of the newer *PKICN25.DLL* files are installed in the:

- Main *PocketBuilder 2.5* directory for use at design time

- *WinCE\arm* directory for deployment to Pocket PC devices
- *WinCE\sparm* directory for deployment to Smartphone devices

If you use an icon from the *oldicons* subdirectory at design time and you deploy your application to a Pocket PC or Pocket PC emulator, you should copy the *PKICN25.DLL* file from the *WinCE\arm\oldicons* directory to your deployment directory or include it in the CAB file that you generate for deployment.

Icon deployment in a generated CAB file
By default, the *PKICN25.DLL* file from the *newicons* subdirectory is included when you build a CAB file from a PocketBuilder project. However, a radio button option on the PocketBuilder Options page of the Enhanced CAB Generation Tool allows you to select the image resource file you want to include in a generated CAB file.

Referencing images in a DLL or EXE file

In PocketBuilder 2.5 you can select a numbered icon or bitmap from a DLL or EXE file and deploy it with your application to a mobile device or emulator. In all visual controls where you can select a BMP file, you can also select a DLL or EXE, although after the DLL or EXE listing, you must type in a semicolon and the number of the image in the file that you want to use.

You can also use the same syntax for selecting an image for DataWindow expression functions, DataWindow object properties, and PowerScript® image property settings and function calls. For example, to set a disabled icon for a picture button with the twelfth icon in a file called iconfile.DLL, you can use the following script:

```
pb_1.DisabledName = "d:\pbhelp\iconfile.DLL;12"
```

Deprecated functionality

Support for x86 devices and x86-based emulators has been removed.