<https://www.ibm.com/docs/en/personal-communications/6.0?topic=programming-introduction-standard-ehllapi-enhanced-ehllapi-winhllapi>

https://www.ibm.com/docs/en/personal-communications/6.0?topic=programming-ehllapi-functions

|  |
| --- |
|  |
| [Table 5. Prerequisite Functions and Associated Dependent Functions](https://www.ibm.com/docs/en/SSEQ5Y_6.0.0/com.ibm.pcomm.doc/books/html/emulator_programming04.htm#FT_Table_5) |

| **Prerequisite Call** | **Functions** | **Access** |
| --- | --- | --- |
| Allocate Communications Buffer (120) | Free Communication Buffer (120) | N/A |
| Connect Window Service (101) | Change PS Window Name (106) Change Switch List Name (105) Disconnect Window Service (102) Query Window Service (103) Window Status (104) | Write Read Query=Read Set=Write Write |
| Connect Presentation Space (1) | Copy Field to String (34) Copy OIA (13) Copy Presentation Space (5) Copy Presentation Space to String (8) Copy String to Field (33) Copy String to Presentation Space (15) Disconnect Presentation Space (2) Find Field Length (32) Find Field Position (31) Query Cursor Location (7) Query Field Attribute (14) Release (12) Reserve (11) Search Field (30) Search Presentation Space (6) Send key (3) Set Cursor (40) Start Playing Macro (110) Wait (4) | Read Read Read Read Write Write Write Read Read Read Read Write Write Read Read Read Write Write Read |
| Connect Structured Field (120) | Disconnect Structured Field (121) Get Request Completion (125) Read Structured Field (126) Write Structured Field (127) | N/A |
| Read Structured Field (126) | Get Request Completion (125) | N/A |
| Start Close Intercept (41) | Query Close Intercept (42) Stop Close Intercept (43) | N/A |
| Start Host Notification (23) | Query Host Update (24) Stop Host Notification (25) |  |
| Start Keystroke Intercept (50) | Get Key (51) Post Intercept Status (52) Stop Keystroke Intercept (53) Send Key (3) if edit keystrokes are to be sent (edit keystroked support is available in Enhanced Mode) | N/A |
| Write Structured Field (127) | Get Request Completion (125) |  |
|  |  |  |

PSID Handling for Functions Not Requiring Connect

Some functions can interact with a host presentation space whether it is connected or not. These functions allow you to specify the PSID in the calling data string parameter. They are as follows:

* **Connect Presentation Space** (1)
* **Convert Position RowCol** (99)
* **Get Key** (51)
* **Post Intercept Status** (52)
* **Query Close Intercept** (42)
* **Query Host Update** (24)
* **Query Session Status** (22)
* **Start Close Intercept** (41)
* **Start Host Notification** (23)
* **Start Keystroke Intercept** (50)
* **Stop Close Intercept** (43)
* **Stop Host Notification** (25)
* **Stop Keystroke Intercept** (53)

All except the first two of these functions allow you to specify the PSID using either:

* A letter *A* through *Z*
* A blank or a null

[Table 3. EHLLAPI Return Codes](https://www.ibm.com/docs/en/SSEQ5Y_6.0.0/com.ibm.pcomm.doc/books/html/emulator_programming04.htm#FT_Table_3)

| **Return Code** | **Explanation** |
| --- | --- |
| 0 | The function successfully executed, or no update since the last call was issued. |
| 1 | An incorrect host presentation space ID was specified. The specified session either was not connected, does not exist, or is a logical printer session. |
| 2 | A parameter error was encountered, or an incorrect function number was specified. (Refer to the individual function for details.) |
| 4 | The execution of the function was inhibited because the target presentation space was busy, in X CLOCK state (X []), or in X SYSTEM state. |
| 5 | The execution of the function was inhibited for some reason other than those stated in return code 4. |
| 6 | A data error was encountered due to specification of an incorrect parameter (for example, a length error causing truncation). |
| 7 | The specified presentation space position was not valid. |
| 8 | A functional procedure error was encountered (for example, use of conflicting functions or missing prerequisite functions). |
| 9 | A system error was encountered. |
| 10 | This function is not available for EHLLAPI. |
| 11 | This resource is not available. |
| 12 | This session stopped. |
| 24 | The string was not found, or the presentation space is unformatted. |
| 25 | Keystrokes were not available on input queue. |
| 26 | A host event occurred. See **Query Host Update** (24) for details. |
| 27 | File transfer was ended by a Ctrl+Break command. |
| 28 | Field length was 0. |
| 31 | Keystroke queue overflow. Keystrokes were lost. |
| 32 | An application has already connected to this session for communications. |
| 33 | Reserved. |
| 34 | The message sent to the host was canceled. |
| 35 | The message sent from the host was canceled. |
| 36 | Contact with the host was lost. |
| 37 | Inbound communication has been disabled. |
| 38 | The requested function has not completed its execution. |
| 39 | Another DDM session is already connected. |
| 40 | The disconnection attempt was successful, but there were asynchronous requests that had not been completed at the time of the disconnection. |
| 41 | The buffer you requested is being used by another application. |
| 42 | There are no outstanding requests that match. |
| 43 | The API was already locked by another EHLLAPI application (on LOCK) or API not locked (on UNLOCK). |
|  | |

### **ASCII Mnemonics**

Keystrokes originating at a host keyboard might have a corresponding ASCII value. The response of the **Get Key** (51) function to a keystroke depends on whether the key is defined and also on whether the key is defined as an ASCII value or an ASCII mnemonic.

The keyboard for one session might not be capable of producing some codes needed by the another session. ASCII mnemonics that represent these codes can be included in the data string parameter of the **Send Key** (3) function.

The capabilities of the **Send Key** (3) function and the **Get Key** (51) function allow sessions to exchange keystrokes that might not be represented by ASCII values or by an available key. A set of mnemonics that can be generated from a keyboard is provided. These mnemonics let you use ASCII characters to represent the special function keys of the workstation keyboard.

Mnemonics for unshifted keys consist of the escape character followed by an abbreviation. This is also true for the shift keys themselves, Upper shift, Alt, and Ctrl. Mnemonics for shifted keys consist of the mnemonic for the shift key followed by the mnemonic for the unshifted key. Hence the mnemonic for a shifted key is a 4-character sequence of escape character, abbreviation, escape character, abbreviation.

The default escape character is @. You can change the value of the escape character to any other character with the ESC=c option of the **Set Session Parameters** (9) function. The following text uses the default escape character, however.

Shift indicators that are not part of the ASCII character set are represented to the host application by 2-byte ASCII mnemonics as follows:

|  |  |
| --- | --- |
| **Upper shift** | @S |
| **Alt** | @A |
| **Ctrl** | @r |

Mnemonics for these shift indicators are never received separately by an application. Likewise, they are never sent separately by an application. Shift indicator mnemonics are always accompanied by a non-shift-indicator character or mnemonic.

The abbreviations used make the mnemonics for special keys easy to remember. An alphabetic key code has been used for the most common keys. For example, the Clear key is C; the Tab key is T, and so on. Please note that the uppercase and lowercase alphabetic characters are mnemonic abbreviations for different keys.

The following text describes the use of these functions.

#### General

All defined keys are represented by either:

* A 1-byte ASCII value that is part of the 256-element ASCII character set, or
* A 2-, 4-, or 6-byte ASCII mnemonic

To represent a key defined as an ASCII character, a 1-byte ASCII value that corresponds to that character is used.

To represent a key defined as a function, a 2-, 4-, or 6-byte ASCII mnemonic that corresponds to that function is used. For example, to represent the backtab key, @B is used. To represent PF1, @1 is used. To represent Erase Input, @A@F is used. See the following lists:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| @B | Left Tab | | @0 | Home | | | @h | PF17 |
| @C | Clear | | @1 | PF1/F1 | | | @i | PF18 |
| @D | Delete | | @2 | PF2/F2 | | | @j | PF19 |
| @E | Enter | | @3 | PF3/F3 | | | @k | PF20 |
| @F | Erase EOF | | @4 | PF4/F4 | | | @l | PF21 |
| @H | Help (PC400) | | @5 | PF5/F5 | | | @m | PF22 |
| @I | Insert | | @6 | PF6/F6 | | | @n | PF23 |
| @J | Jump | | @7 | PF7/F7 | | | @o | PF24 |
| @L | Cursor Left | | @8 | PF8/F8 | | | @q | End |
| @N | New Line | | @9 | PF9/F9 | | | @u | Page UP (PC400) |
| @O | Space | | @a | PF10/F10 | | | @v | Page Down (PC400) |
| @P | Print | | @b | PF11/F11 | | | @x | PA1 |
| @R | Reset | | @c | PF12/F12 | | | @y | PA2 |
| @T | Right Tab | | @d | PF13 | | | @z | PA3 |
| @U | Cursor Up | | @e | PF14 | | | @@ | @ (at) symbol |
| @V | Cursor Down | | @f | PF15 | | | @$ | Alternate Cursor |
| @X | DBCS | | @g | PF16 | | | @< | Backspace |
| @Z | Cursor Right | |  |  | | |  |  |
| @A@C | | Test (PC400) | | | @A@e | Pink (PC/3270) | | |
| @A@D | | Word Delete | | | @A@f | Green (PC/3270) | | |
| @A@E | | Field Exit | | | @A@g | Yellow (PC/3270) | | |
| @A@F | | Erase Input | | | @A@h | Blue (PC/3270) | | |
| @A@H | | System Request | | | @A@i | Turquoise (PC/3270) | | |
| @A@I | | Insert Toggle | | | @A@j | White (PC/3270) | | |
| @A@J | | Cursor Select | | | @A@l | Reset Host Color (PC/3270) | | |
| @A@L | | Cursor Left Fast | | | @A@t | Print (Personal Computer) | | |
| @A@Q | | Attention | | | @A@u | Rollup (PC400) | | |
| @A@R | | Device Cancel | | | @A@v | Rolldown (PC400) | | |
| @A@T | | Print Presentation Space | | | @A@y | Forward Word Tab | | |
| @A@U | | Cursor Up Fast | | | @A@z | Backward Word Tab | | |
| @A@V | | Cursor Down Fast | | | @A@- | Field - (PC400) | | |
| @A@Z | | Cursor Right Fast | | | @A@+ | Field + (PC400) | | |
| @A@9 | | Reverse Video | | | @A@< | Record Backspace (PC400) | | |
| @A@b | | Underscore (PC/3270) | | | @S@E | Print Presentation Space on Host (PC400) | | |
| @A@c | | Reset Reverse Video (PC/3270) | | | @S@x | Dup | | |
| @A@d | | Red (PC/3270) | | | @S@y | Field Mark | | |

**Notes:**

1. The first @ symbol in the first table represents the escape character. The first and second @ symbol in the second table is the escape character. The @ symbol is the default escape character. You can change the value of the escape character using the ESC=c option of the **Set Session Parameters** (9) function.

If you change the escape character to #, the literal sequences used to represent the Backtab, Home, and Erase Input keys become #B, #0, and #A#F, respectively.

Also, the literal sequence used to represent the @ symbol becomes #@.

1. If you send the mnemonic for print screen (that is, either @P or @A@T), place it at the end of the calling data string.
2. If you send the mnemonic for device cancel (that is, @A@R), it is passed through with no error message; however, local copy is not stopped.

#### Get Key (51) Function

If the terminal operator types a key defined as an ASCII character, the host application receives a 1-byte ASCII value that corresponds to that character.

If the operator types a key defined as a function, the host application receives a 2-, 4-, or 6-byte ASCII mnemonic that corresponds to that function. For example, if the **Backtab** key is typed, @B is received. If **PF1** is pressed, @1 is received. If **Erase Input** is pressed, @A@F is received.

If the operator types a defined shift key combination, the host application receives the ASCII character, or the 2-, 4-, or 6-byte ASCII mnemonic that corresponds to the defined character or function.

If the operator types an individual key that is not defined, the **Get Key** (51) function returns a return code of 20 and nothing is sent to the host application.

The **Get Key** (51) function prefixes all characters and mnemonics sent to the host application with two ASCII characters. The first ASCII character is the PSID of the host presentation space to which the keystrokes are sent. The other character is an A, S, or M for ASCII, special shift, or mnemonic, respectively. See [Return Parameters](https://www.ibm.com/docs/en/SSEQ5Y_6.0.0/com.ibm.pcomm.doc/books/html/emulator_programming08.htm#spot51).

#### Send Key (3) Function

To send an ASCII character to another session, include that character in the data string parameter of the **Send Key** (3) function.

To send a function key to another session, include the ASCII mnemonic for that function in the data string parameter of the **Send Key** (3) function.

If the **Send Key** (3) function sends an unrecognized mnemonic to the host session a return code rejecting the key might result.

1. The value entered in the calling length parameter is the maximum number of half-second intervals that the **Pause** function waits. For a pause of 20 seconds, a hex value of 0028 (decimal 40) must be passed in the calling length parameter.

The **Wait** function checks the status of the host-connected presentation space. If the session is waiting for a host response (indicated by XCLOCK (X []) or XSYSTEM), the **Wait** function causes EHLLAPI to wait up to 1 minute to see if the condition clears.