Events and Status Codes

This chapter describes events received and status codes returned by Tcl IVR scripts. This chapter includes the following topics:

Events

The following events can be received by the Tcl IVR script. Any events received that are not included below are ignored.

Event	Description
ev_accounting_status_ind	Received when the method list or server group is marked unreachable. The accounting status and method list can be derived using infotag get evt_aaa_status_info [attribute-name].
ev_address_resolved	List of endpoint addresses.
ev_alert	An intermediate event generated by the **leg setup** or **leg setup_continue** commands to set up a call. If specified in the callinfo parameter, *notifyEvents,* the script receives an **ev_alert** message once the destination endpoint is successfully alerted. The script running in the transferee gateway could then disconnect the leg towards the transferring endpoint. If this event is an intercepted event, the application needs to use the **leg setup_continue** command to allow the system to continue with the setup.
ev_any_event	A special wildcard event that can be used in the state machine to represent any event that might be received by the script.
ev_authorize_done	Confirms the completion of the **aaa authorize** command. You can use the evt_status info-tag to determine the authorization status (whether it succeeded or failed).
ev_authenticate_done	Confirms the completion of the authentication command. You can use the evt_status info-tag to determine the authentication status (whether it succeeded or failed).
ev_call_timer0	Indicates that the call-level timer expired.

Event	Description
ev_collectdigits_done	Confirms the completion of the **leg collectdigits** command on the call leg. You can then use the evt_status info-tag to determine the status of the command completion. You can use the evt_dcdigits info-tag to retrieve the collected digits.
ev_connected	An intermediate event generated by the **leg setup** or **leg setup_continue** commands to set up a call. If the callinfo paramater, *notifyEvents*, is specified, the script receives an *ev_connected* message when the system receives a connect event from the destination switch. If this event is an intercepted event, the application needs to use the **leg setup_continue** command to allow the system to continue with the setup.
ev_consult_request	Indicates a call-transfer consultation-id request from an endpoint.
ev_consult_response	Indicates a response to the **leg consult request** command. For return codes, see Consult Status under Status Codes.
ev_consultation_done	Indicated the completion of a **leg consult response** command. For return codes, see Consult Response under Status Codes.
ev_create_done	Confirms the completion of the **connection create** command. You can use the evt_connection info-tag to determine the ID of the completed connection.
ev_destroy_done	Confirms the completion of the **connection destroy** command. You can use the evt_connection info-tag to determine the ID of the connection that was destroyed.
ev_digit_end	Indicates that a digit key is pressed and released. You can use the evt_digit info-tag to determine which digit was pressed. You can use the evt_digit_duration info-tag to determine how long (in seconds) the digit was pressed. This can be used to detect long pounds or long digits.
ev_disconnect_done	Indicates that the call leg has been cleared.
ev_disconnected	Indicates that one of the call legs needs to disconnect. On receiving this event, the script must issue a **leg disconnect** on that call leg. You can use the evt_legs info-tag to determine which call leg disconnected.
ev_disc_prog_ind	Indicates a DISC/PI message is received at a call leg.
ev_facility	Indicates a response to a **leg facility** command.
ev_feature	Indicates a feature event received by the script. The script can use the **set evt_feature_report** information tag to enable or disable the feature events to be intercepted. When the script receives an *ev_feature* event, it can use the **get evt_feature_type** information tag to retrieve the feature type string.
ev_grab	Indicates that an application that called this script is requesting that the script return the call leg. The script receiving this event can clean up and return the leg with a **handoff return** command. Whether this is done is at the discretion of the script receiving the ev_grab event.

Event	Description
ev_hookflash	Indicates a hook flash (such as a quick onhook-offhook in the middle of a call), assuming that the underlying platform or interface supports hook flash detection. It is received by the TCL script when the user presses a hookflash.
ev_handoff	Indicates that the script received one or more call legs from another application. When the script receives this event, you can use the evt_legs and the evt_connections info-tags to obtain a list of the call legs and connection IDs that accompanied the ev_handoff event.
ev_leg_timer	Indicates that the leg timer expired. You can use the evt_legs info-tag to determine which leg timer expired.
ev_media_activity	Indicates the detection of an active call. It is generated when the RTP and RTCP packets are transmitted again after a period of media inactivity.
ev_media_done	Indicates that the prompt playout either completed or failed. You can use the evt_status info-tag to determine the completion status.
ev_media_inactivity	Indicates the detection of an inactive call. It is generated if the RTP and RTCP packets are not received during a specified time period. The time period is specified by the CLI **ip rtcp report interval** and **timer receive-rtcp**.
ev_named_timer	Received when a named_timer expires. The name of the named_timer can be derived by using the *get evt_timer_name* information tag.
ev_proceeding	An intermediate event generated by the **leg setup** or **leg setup_continue** commands to set up a call. If the callinfo paramater, *notifyEvents*, is specified, the script receives an *ev_proceeding* message when the system receives a proceeding event from the remote end. If this event is an intercepted event, the application needs to use the **leg setup_continue** command to allow the system to continue with the setup.
ev_progress	An intermediate event generated by the **leg setup** or **leg setup_continue** commands to set up a call. If the callinfo paramater, *notifyEvents*, is specified, the script receives an *ev_progress* message when the system receives a progress event from the destination switch. If this event is an intercepted event, the application needs to use the **leg setup_continue** command to allow the system to continue with the setup.
ev_returned	Indicates that a call leg that was sent to another application (using **handoff callappl**) has been returned. This event can be accompanied by one or more call legs that were created by the called application. When the script receives this event, you can use the evt_legs and the evt_connections info-tags to obtain a list of the call legs and connection IDs that accompanied the ev_returned event. You can use the evt_iscommand_done info-tag to verify that all of the call legs sent have been accounted for, meaning that the **handoff callappl** command is complete.

Event	Description
ev_setup_done	Indicates that the **leg setup** command has finished. You can then use the evt_status info-tag to determine the status of the command completion (whether the call was successfully set up or failed for some reason).
ev_setup_indication	Indicates that the system received a call. This event and the ev_handoff event are the events that initiate an execution instance of a script.
ev_synthesizer	Indicates the completion of a **media play** command.

ev_tone_detected	Signifies the detection of the requested tone. This event is generated, at most, once after a **leg tonedetect enable** command is issued. Tone detection is disabled after this event arrives. Use the *evt_status* information tag to determine the detected tone. See Status Codes, page 5-6, for possible tone detect status values.
	Example:
	`set fsm(WAIT_FOR_CNG, ev_tone_detected) "act_process_td_event, same_state")`
	`proc act_process_td_event { } { set Tone1 [infotag get evt_status] if { $Tone1 == "td_003" } # Do stuff here }`

ev_transfer_request	Indicates a call transfer from an endpoint to the application.
ev_transfer_status	An intermediate event generated by the **leg setup** command. If specified in the callinfo parameter, *notifyEvents*, the script receives an **ev_trasfer_status** message. The *ev_status* information tag would then contain the status value of the call transfer.
ev_vxmldialog_done	Received when the VXML dialog completes. This could be because of a VXML dialog executing an <exit/> tag or interpretation completing the current document without a transition to another document. The dialog could also complete due to an interpretation failure or a document error. This completion status is also available through the evt_status info-tag.
ev_vxmldialog_event	Received by the Tcl IVR application when the VXML dialog initiated on a leg executes a sendevent object tag. The VXML subevent name is available through the evt_vxmlevent info-tag. All events thrown from the dialog markup are of the form vxml.dialog.*. All events generated by the system—perhaps as an indirect reaction to the VXML document executing a certain tag or throwing a certain event—like the dialog completion event are of the form vxml.session.*.
ev_msg_indication	Signals an incoming message.
ev_session_indication	Signals the start of a new session.
ev_session_terminate	Stop the current TCL session.
ev_subscribe_done	Subscription request completed.
ev_unsubscribe_done	Unsubscribe request completed.
ev_unsubscribe_indication	Server terminated the subscription.
ev_notify	Notify indication received.
ev_notify_done	Notification request completed

Event	Description
ev_subscribe_cleanup	Received when a 'clear subscription <session id | all' CLI is executed.
ev_returned	Received when another application instance returns the call leg.
ev_grab	If a user stops a TCL session that has already handed a call off to a second session, the session sends an ev_grab event to the second session. If the second session returns the call leg, the first session cleans up. If the second session does not return the call leg, the first session stops executing, but does not clean up completely until the second session disconnects the call leg, or returns it.

Usage Notes:

- Scripts must check the return status for the *ev_subscribe_done* event. This event indicates that a response is received from server. A return code of **su**_*000* indicates that a positive response has been received. A return code of **su**_*002* or above indicates a negative response from the server.
- A subscription is complete only when an *ev_subscribe_done* event and the first notification from server are received. An application should close its instances only after making sure the subscription is complete.
- The script receives an *ev_unsubscribe_indication* event when the server terminates the subscription. The script can access header and content information associated with this event.
- If the subscription timer expires, the script receives an *ev_unsubscribe_indication* event with a status code of **ui**_*003*. Since this is an internal event, there is no header or content information associated with this event.
- When an *ev_subscribe_cleanup* event is received, the script must close the subscription. If no response is received within 5 seconds, the infrastructure closes the subscription. Make sure the script handles this event.
- If the instance making the subscription is already closed and an *ev_notify_indication*, *ev_subscribe_cleanup*, or *ev_unsubscribe_indication* event is received, a new instance is created and the event is handed to it.

Status Codes

The evt_status info-tag returns a status code for the event received. This sections lists the possible status codes and their meaning.

Status codes are grouped according to function. The first two characters of the status code indicate the grouping.

- au—Authentication status
- ao—Authorization status
- cd—Digit collection status
- cr—Consult response
- cs—Consult status
- di— Disconnect cause
- fa—Facility
- ft—Feature type
- lg—Leg state status
- ls—Leg setup status
- ms—Media status
- td—Tone detect
- ts—Transfer status
- vd—Voice dialog completion status

Authentication Status

Authentication status is reported in **au**_*xxx* format:

Value for *xxx*	Description
000	Authorization was successful.
001	Authorization error.
002	Authorization failed.

Authorization Status

Authorization status is reported in **ao**_*xxx* format:

Value for *xxx*	Description
000	Authorization was successful.
001	Authorization error.
002	Authorization failed.

Digit Collection Status

Digit collection status is reported in **cd**_*xxx* format:

Value for *xxx*	Description
001	The digit collection timed out, because no digits were pressed and not enough digits were collected for a match.
002	The digit collection was aborted, because the user pressed an abort key.
003	The digit collection failed, because the buffer overflowed and not enough digits were collected for a match.
004	The digit collection succeeded with a match to the dial plan.
005	The digit collection succeeded with a match to one of the patterns.
006	The digit collection failed because the number collected was invalid.
007	The digit collection was terminated because an ev_disconnected event was received on the call leg.
008	The digit collection was terminated because an ev_grab event was received on the call leg.
009	The digit collection successfully turned on digit reporting to the script.
010	The digit collection was terminated because of an unsupported or unknown feature or event.

Consult Response

Feature type is reported in **cr**_*xxx* format:

Value for *xxx*	Description
000	Success
001	Failed, invalid state
002	Failed, timeout
003	Failed, abandon
004	Failed, protocol error

Consult Status

Feature type is reported in **cs**_*xxx* format:

Value for *xxx*	Description
000	Consultation success, consult-id available
001	Consultation failed, request timeout
002	Consultation failed
003	Consultation failed, request rejected
004	Consultation failed, leg disconnected
005	Consultation failed, operation unsupported

Disconnect Cause

Disconnect causes use the format **di**_*xxx* where *xxx* is the Q931 cause code. Possible values are:

Value for *xxx*	Description
000	Uninitialized
001	Unassigned number
002	No route to the transit network
003	No route to the destination
004	Send information tone
005	Misdialed trunk prefix
006	Unacceptable channel
007	Call awarded
008	Preemption
009	Preemption reserved
016	Normal
017	Busy
018	No response from the user
019	No answer from the user
020	Subscriber is absent
021	Call rejected
022	Number has changed
026	Selected user is clearing
027	Destination is out of order
028	Invalid number
029	Facility rejected
030	Response to status inquiry
034	No circuit available
035	Requested VPCI VCI is not available
036	VPCI VCI assignment failure
037	Cell rate is not available
038	Network is out of order
039	Permanent frame mode is out of service
040	Permanent frame mode is operational
041	Temporary failure
042	Switch is congested
043	Access information has been discarded
044	No required circuit
045	No VPCI VCI is available
046	Precedence call blocked

Value for *xxx*	Description
047	No resource available
048	DSP error
049	QoS is not available
050	Facility is not subscribed
053	Outgoing calls barred
055	Incoming calls barred
057	Bearer capability is not authorized
058	Bearer capability is not available
062	Inconsistency in the information and class
063	Service or option not available
065	Bearer capability is not implemented
066	Change type is not implemented
069	Facility is not implemented
070	Restricted digital information only
079	Service is not implemented
081	Invalid call reference value
082	Channel does not exist
083	Call exists and call ID in use
084	Call ID in use
085	No call suspended
086	Call cleared
087	User is not in CUG
088	Incompatible destination
090	CUG does not exist
091	Invalid transit network
093	AAL parameters not supported
095	Invalid message
096	Mandatory information element (IE) is missing
097	Message type is not implemented
098	Message type is not compatible
099	IE is not implemented
100	Invalid IE contents
101	Message in incomplete call state
102	Recovery on timer expiration
103	Nonimplemented parameter was passed on
110	Unrecognized parameter message discarded
111	Protocol error

Value for *xxx*	Description
127	Internetworking error
128	Next node is unreachable
129	Holst Telephony Service Provider Module (HTSPM) is out of service
160	DTL transit is not my node ID

Facility

Leg setup requesting address resolution status is reported in **fa**_*xxx* format:

Value for *xxx*	Description
000	supplementary service request succeeded
003	supplementary service request unavailable
007	supplementary service was invoked in an invalid call state
009	supplementary service was invokes in a non-incoming call leg
010	supplementary service interaction is not allowed
050	MCID service is not subscribed
051	MCID request timed out
052	MCID is not configured for this interface
053	Unknown error
054	Initialization error

Feature Type

Feature type is reported in **ft**_*xxx* format:

Value for *xxx*	Description
001	Fax
002	Modem
003	Modem_phase
004	Hookflash
005	OnHook
006	OffHook

Leg State

The state of the call leg and the corresponding status code is reported in **lg**_***xxx*** format:

Value for *xxx*	Call Leg
000	LEG_INIT
001	LEG_INCOMING_FIRST

Value for *xxx*	Call Leg
001	LEG_INCINIT
002	LEG_INCACKED
003	LEG_INCPROCEED
004	LEG_ALERTING
005	LEG_INCCONNECTED
006	LEG_INCDISCONNECTING
006	LEG_INCOMING_LAST
007	LEG_OUTGOING_FIRST
007	LEG_OUTINIT
008	LEG_OUTPROCEED
009	LEG_OUTRINGING
010	LEG_OUTCONNECTED
011	LEG_OUTDISCONNECTING
011	LEG_OUTGOING_LAST
012	LEG_DISCONNECTED

Leg Setup Status

Leg setup status is reported in **ls**_*xxx* format:

Value for *xxx*	Description
000	The call is active or was successful.
001	The outgoing call leg was looped.
002	The call setup timed out (meaning that the destination phone was alerting, but no one answered). The limit of this timeout can be specified in the **leg setup** command.
003	The call setup failed because of a lack of resources in the network.
004	The call setup failed because of an invalid number.
005	The call setup failed for reasons other than a lack of resources or an invalid number.
006	Unused; setup failure.
007	The destination was busy.
008	The incoming side of the call disconnected.
009	The outgoing side of the call disconnected.
010	The conferencing or connecting of the two call legs failed.
011	Supplementary services internal failure
012	Supplementary services failure
013	Supplementary services failure. Inbound call leg was disconnected.
014	The call was handed off to another application.
015	The call setup was terminated by an application request.

Value for *xxx*	Description
016	The outgoing called number was blocked.
018	• Called number is blocked • No service available (Q.850 cause 63)
026	Leg redirected
031	Transfer request acknowledge
032	Transfer target alerting (future SIP use)
033	Transfer target trying (future SIP use)
040	Transfer success
041	Transfer success with transfer-to party connected (SIP only)
042	Transfer success unacknowledged (SIP only)
050	Transfer fail
051	Transfer failed, bad request (SIP only)
052	Transfer failed, destination busy
053	Transfer failed, request cancelled
054	Transfer failed, internal error
055	Transfer failed, not implemented (SIP only)
056	Transfer failed, service unavailable or unsupported
057	Transfer failed, leg disconnected
058	Transfer failed, multiple choices (SIP only)
059	Transfer failed, timeout; no response to transfer request

Media Status

Media status is reported in **ms**_*xyy* format:

***x* indicates the command**		***yy* indicates the status of the command**	
Value for *x*	**Description**	**Value for *yy***	**Description**
0	Status for a **media play** command.	00	The command was successful and the prompt finished.[1]
1	Status for a media record command.	01	Failure
2	Status for a **media stop** command.	02	Unsupported feature or request
3	Status for a **media pause** command.	03	Invalid host or URL specified
4	Status for a **media resume** command.	04	Received disconnected
5	Status for a **media seek** command to forward.	05	The prompt was interrupted by a key press.
6	Status for a **media seek** command to rewind.		

1. Valid for the **media play** command only, because media_done events are not received for successful completion of other media commands.

Subscribe/Notify

The following return codes are defined for Subscribe/Notify events:

- ev_subscribe_done **su**_*xxx*
- ev_notify_done **no**_*xxx*
- ev_unsubscribe_done **us**_*xxx*
- ev_unsubscribe_ind **ui**_*xxx*

where *xxx* in the strings above represent the following:

Value for *xxx*	Description
000	Success
001	Pending
002	Generic failure
003	Subscription expired
004	Socket error
005	DNS error
006	Request timed-out error
007	Connection timed-out error
008	Connection create failed
009	Internal error
010	Response error
099	Undefined

Tone Detect

Tone detect is reported in **td**_*xxx* format:

Value for *xxx*	Description
000	Invalid inband signal
001	FAX_V21
002	FAX_CED
003	FAX_CNG
004	MODEM_2100HZ
005	MODEM_2100HZ_PHASE
006	VOICE_SILENCE
007	CP_TONE

Transfer Status

Transfer status is reported in **ts**_*xxx* format:

Value for *xxx*	Description
000	Generic transfer success
001	Transfer success, transfer-to party is alerting
002	Transfer success, transfer-to party is answered
003	Transfer finished; however, the result of the transfer is not guaranteed
004	Transfer request is accepted
005	Transferee is trying to reach transfer-to party
006	Transfer request is rejected by transferee
007	Invalid transfer number
008	Transfer-to party unreachable
009	Transfer-to party is busy

VoiceXML Dialog Completion Status

VoiceXML dialog completion status is reported in **vd**_*xxx* format:

Value for *xxx*	Description
000	Normal completion because of the <exit> tag or execution reaching the end of the document.
001	Termination because of the default VXML event handling requiring VXML termination.
002	Terminated by the Tcl IVR application.
003	Internal failure.