# Tencent Voice Service Reference

## Alert Interface

The Alerts Interface exposes directives and events that are used to set, manage and cancel timers, alarms, and reminders. Your client must implement the logic necessary to manage timers, alarms, and reminders if internet connectivity is lost or if the on-product clock is out of sync with NTP.

### State Diagram

The following diagram illustrates state changes driven by the Alerts component. Boxes represent Alerts states and connectors indicate transitions.

Alerts supports the following states:

****IDLE:**** Prior to a previously set alert going off, the Alerts component should be in the *idle* state. Alerts should also return to an *idle* state once an alert is stopped/finished. This can occur as the result of user speech, a physical button press, or GUI affordance.

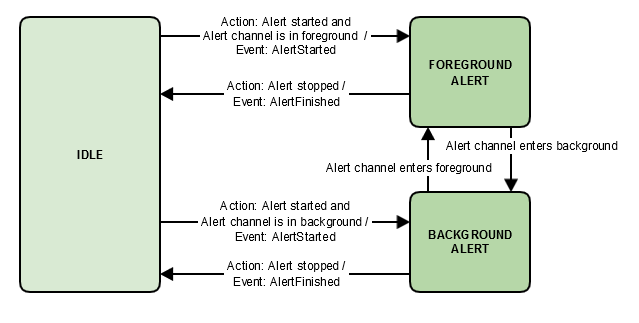
****FOREGROUND ALERT:**** Assuming an on-client alert has already been set, Alerts should transition from the *idle*state to the *alert foreground* state when an alert starts and the AlertStarted event is sent to the Tencent Voice Service.

This is only true if the Alerts channel is in the foreground, which implies the Dialog channel is inactive.

When an alert is stopped via speech, button press, or GUI affordance, the Alerts component should transition from the *alert foreground* state to the *idle* state.

If the Dialog channel becomes active while an alert is going off, your Alerts component should transition from the *foreground alert* state to the *background alert* state as long as the Dialog channel is active. When the Dialog channel becomes inactive, it should return to the *foreground alert* state until it is stopped/finished.

****BACKGROUND ALERT:**** The Alerts component should only transition to the *background alert* state when the Dialog channel is active.



### **Alerts Context**

TVS expects a client to report the status of all locally stored alerts with each event that requires context. Alerts are organized into two lists: allAlerts and activeAlerts. allAlerts is a complete list of locally stored alerts. activeAlerts is a list of alerts currently in focus or sounding for an end user.

****Sample Message****

{

"header": {

"namespace": "Alerts",

"name": "AlertsState"

},

"payload": {

"allAlerts": [

{

"token": "{{STRING}}",

"type": "{{STRING}}",

"scheduledTime": "{{STRING}}"

}

],

"activeAlerts": [

{

"token": "{{STRING}}",

"type": "{{STRING}}",

"scheduledTime": "{{STRING}}"

}

]

}

}

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| allAlerts | Key/value pairs for allAlerts | object |
| allAlerts.token | Alert token returned by TVS when the alert was set. | string |
| allAlerts.type | Identifies the alert type.  ****Accepted Values****: TIMER, ALARM, REMINDER. | string |
| allAlerts.scheduledTime | Time the alert is scheduled in ISO 8601 format. | string |
| activeAlerts | Key/value pairs for activeAlerts | object |
| activeAlerts.token | The token for the alert that is currently firing. | string |
| activeAlerts.type | Identifies the alert type.  ****Accepted Values****: TIMER or ALARM | string |
| activeAlerts.scheduledTime | Time the alert is scheduled in ISO 8601 format. | string |

### **SetAlert Directive**

This directive instructs your client to set a timer, alarm, or reminder for a specific duration or time. Your client may receive the SetAlert directive as a result of a speech request to set an alert.

If loopCount is absent from the payload, the alert must sound for one hour or until it is stopped by the user (voice request or physical affordance).

Cloud provided assets take precedence over locally stored audio files. If assets are provided they must be played for the user in the order provided in the assetPlayOrder list. Otherwise, use the audio files for alerts provided by Tencent.

**Important:** If the assets.url[i] is unreachable, or if your client fails to download the associated files, it should play the audio files for the associated alert type provided by Tencent and adhere to the provided loopCount and loopPauseInMilliSeconds.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Alerts",

"name": "SetAlert",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"type": "{{STRING}}",

"scheduledTime": "{{STRING}}",

"assets": [

{

"assetId": "{{STRING}}",

"url": "{{STRING}}"

},

{

"assetId": "{{STRING}}",

"url": "{{STRING}}"

},

],

"assetPlayOrder": [ {{LIST}} ],

"backgroundAlertAsset": "{{STRING}}",

"loopCount": {{LONG}},

"loopPauseInMilliSeconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token that uniquely identifies the alert. | string |
| type | Identifies the alert type. If an unrecognized value is sent to your client, it should default to an ALARM.  ****Accepted values:**** TIMER, ALARM, REMINDER. | string |
| scheduledTime | The scheduled time for an alert in ISO 8601 format. | string |
| assets | A list that contains audio assets to be played to the user. | list |
| assets[i].assetId | A unique identifier for the audio asset. | string |
| assets[i].url | Identifies the location of the asset in the cloud. This asset may be downloaded and cached by your client. The URL provided is valid for 60 minutes after the scheduledTime. | string |
| assetPlayOrder | The sequence that audio assets must be played. The list is comprised of assetIds.   ****Note****: i) assetIds may appear multiple times in the list. When this occurs, all assetIds must be played. ii) If your client fails to download and cache the assets, your device should use the audio files provided by Tencent. | list |
| backgroundAlertAsset | If present, the backgroundAlertAsset value will match an assetId in the assets list. If backgroundAlertAsset is not included in the payload, default to the Tencent provided sound for ****TVS****. | string |
| loopCount | The number of times each sequence of assets must be played. For example: If the value is 2, your client must loop through assetPlayOrder two times.  ****Note****: If loopCount is absent from the payload, you must loop the assets for one hour, or until the alert is stopped by the user. | long |
| loopPauseInMilliSeconds | Pause duration between each asset loop. For example: If the loopPauseInMilliSeconds is 300 and the loopCount is 3, your client must pause for 300 millisecond between each asset loop.  ****Note****: If this value is not specified or is set to 0, there must not be any pause between asset loops. | long |

### **SetAlertSucceeded Event**

The SetAlertSucceeded event must be sent to TVS after receiving a [SetAlert](https://developer.amazon.com/zh/docs/alexa-voice-service/alerts.html" \l "setalert) directive, when the client successfully sets the alert.

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "SetAlertSucceeded",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the SetAlert directive. | string |

### **SetAlertFailed Event**

The SetAlertFailed event must be sent to TVS after receiving a [SetAlert](https://developer.amazon.com/zh/docs/alexa-voice-service/alerts.html" \l "setalert) directive, when the client fails to sets an alert.

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "SetAlertFailed",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

| ****Parameter**** | ****Description**** | ****Type**** |
| --- | --- | --- |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

| ****Parameter**** | ****Description**** | ****Type**** |
| --- | --- | --- |
| token | An opaque token provided by the SetAlert directive. | string |

### **DeleteAlert Directive**

This directive is sent from TVS instructing your client to delete an existing alert. Your client may receive the DeleteAlert directive as a result of a speech request to cancel/delete a timer, alarm, or reminder.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Alerts",

"name": "DeleteAlert",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token that uniquely identifies the alert. | string |

### **DeleteAlertSucceeded Event**

The DeleteAlertSucceeded event must be sent to TVS after receiving a [DeleteAlert](https://developer.amazon.com/zh/docs/alexa-voice-service/alerts.html" \l "deletealert) directive, when the client successfully deletes or cancels an existing alert.

**Note:** For more information on when to send the DeleteAlertSucceeded event, please see [Alerts Overview](https://developer.amazon.com/docs/alexa-voice-service/alerts-overview.html).

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "DeleteAlertSucceeded",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the DeleteAlert directive. | string |

### **DeleteAlertFailed Event**

The DeleteAlertFailed event must be sent to TVS after receiving a [DeleteAlert](https://developer.amazon.com/zh/docs/alexa-voice-service/alerts.html" \l "deletealert) directive, when the client fails to delete or cancel an existing alert.

**Note:** For more information on when to send the DeleteAlertFailed event, please see [Alerts Overview](https://developer.amazon.com/docs/alexa-voice-service/alerts-overview.html).

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "DeleteAlertFailed",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the DeleteAlert directive. | string |

### **DeleteAlerts Directive**

This directive instructs your client to delete all existing alerts on a product. Each alert is identified by a unique token within the payload. Your client may receive the DeleteAlerts directive as a result of a speech request to cancel/delete all alerts.

If one or more alerts fail to be deleted, the client must rollback and send a [DeleteAlertsFailed](https://developer.amazon.com/zh/docs/alexa-voice-service/DeleteAlertsFailed-Event) event to TVS. TVS will then retry until all alerts are deleted, at which point a [DeleteAlertsSucceeded](https://developer.amazon.com/zh/docs/alexa-voice-service/deletealertssucceeded) event must be sent by your client.

**Note:** If one or more alert tokens are not found on the product, the client should proceed with deleting all matching tokens. In this case, this process *would not* fail. DeleteAlerts should be sent only if one or more existing alert tokens on the product fail to be deleted.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Alerts",

"name": "AlertEnteredBackground",

"messageId": "{{STRING}}"

},

"payload": {

"tokens": ["{{STRING}}",...]

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| tokens | An array of tokens. Each token is a string that uniquely represents an alert on the product. | string |

### **DeleteAlertsSucceeded Event**

The DeleteAlertsSucceeded event must be sent to TVS after receiving a [DeleteAlerts](https://developer.amazon.com/zh/docs/alexa-voice-service/alerts.html" \l "deletealerts) directive, when the client successfully deletes or cancels all existing alerts within the tokens array.

**Note:** For more information on when to send the DeleteAlertsSucceeded event, please see [Alerts Overview](https://developer.amazon.com/docs/alexa-voice-service/alerts-overview.html).

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "AlertEnteredBackground",

"messageId": "{{STRING}}"

},

"payload": {

"tokens": ["{{STRING}}",...]

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| tokens | An array of tokens. Each token is a string that uniquely represents an alert on the product. | string |

### **DeleteAlertsFailed Event**

The DeleteAlertsFailed event must be sent to TVS after receiving a [DeleteAlerts](https://developer.amazon.com/zh/docs/alexa-voice-service/alerts.html" \l "deletealerts) directive, when the client fails to delete or cancel at least one of the existing alerts within the tokens array.

**Note:** For more information on when to send the DeleteAlertsFailed event, please see [Alerts Overview](https://developer.amazon.com/docs/alexa-voice-service/alerts-overview.html).

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "AlertEnteredBackground",

"messageId": "{{STRING}}"

},

"payload": {

"tokens": ["{{STRING}}",...]

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| tokens | An array of tokens. Each token is a string that uniquely represents an alert on the product. | string |

### **AlertStarted Event**

The AlertStarted event must be sent to TVS when an alert is triggered at its scheduled time.

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "AlertStarted",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the SetAlert directive. | string |

### **AlertStopped Event**

The AlertStopped event must be sent to TVS when an active alert is stopped. An alert is stopped when:

1. A [DeleteAlert](https://developer.amazon.com/zh/docs/alexa-voice-service/alerts.html" \l "deletealert) directive is received. After sending an AlertStopped event, your client must inform TVS if the alert was successfully deleted with either a [DeleteAlertSucceeded](https://developer.amazon.com/zh/docs/alexa-voice-service/alerts.html" \l "deletealertsucceeded) event or [DeleteAlertFailed](https://developer.amazon.com/zh/docs/alexa-voice-service/alerts.html" \l "deletealertfailed) event. This interaction is illustrated in [Alerts Overview](https://developer.amazon.com/docs/alexa-voice-service/alerts-overview.html).

2. A physical control (hardware button or GUI) is used to stop the alert.

3. The loopCount is complete, or an alert without a loopCount has played for an hour and is stopped locally.

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "AlertStopped",

"messageId": "{STRING}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the SetAlert directive. | string |

### **AlertEnteredForeground Event**

The AlertEnteredForeground event must be sent from your client to TVS when an active alert enters the foreground (plays at full volume) or re-enters the foreground after a concurrent interaction on the Dialog channel finishes.

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "AlertEnteredForeground",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the SetAlert directive. | string |

### **AlertEnteredBackground Event**

The AlertEnteredBackground event must be sent from your client to TVS when an active alert exits the foreground (attenuates or pauses) while a concurrent interaction on the Dialog channel is occurring.

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "AlertEnteredBackground",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided in the SetAlert directive. | string |

**SetVolume Directive**

This directive instructs a client to set the absolute volume level for an alert.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Alerts",

"name": "SetVolume",

"messageId": "{{STRING}}",

"dialogRequestId": {{STRING}}

},

"payload": {

"volume": "{{LONG}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| volume | The absolute volume level scaled from 0 (min) to 100 (max).  ****Accepted values:**** Any value between 0 and 100, inclusive. | long |

### **AdjustVolume Directive**

This directive instructs a client to adjust the relative volume level of an alert.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Alerts",

"name": "AdjustVolume",

"messageId": "{{STRING}}",

"dialogRequestId": {{STRING}}

},

"payload": {

"volume": "{{LONG}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| volume | The relative volume adjustment. A positive or negative long value that is used to increase or decrease volume in relation to the current volume setting.  ****Accepted values:**** Any value between -100 and 100, inclusive. | long |

### **VolumeChanged Event**

This event must be sent to TVS after receiving either a SetVolume or AdjustVolume directive.

****Sample Message****

{

"event": {

"header": {

"namespace": "Alerts",

"name": "VolumeChanged",

"messageId": "{{STRING}}"

},

"payload": {

"volume": "{{LONG}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| volume | This is the volume that is locally adjusted on your product.  ****Accepted values:**** Volume must be a value between 0 (min) and 100 (max), inclusive.  ****Important:**** If your product locally supports volume adjustment from 0 to 10, you must scale this to a range of 0 to 100. For example, when the user increases the volume to 8, TVS expects the volume value sent to be 80. | long |

## AudioPlayer interface

The AudioPlayer interface provides directives for controlling audio playback through voice, and events to manage and monitor playback progression. If you are looking to map playback controls to buttons (physical or GUI), please reference the [PlaybackController interface](https://developer.amazon.com/docs/alexa-voice-service/playbackcontroller.html).

### **State Diagram**

The following diagram illustrates state changes driven by AudioPlayer components. Boxes represent AudioPlayer states and the connectors indicate state transitions.

AudioPlayer has the following states:

****IDLE:**** AudioPlayer is only in an *idle* state when a product is initially powered on or rebooted and prior to acting on a [Play](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "play) directive.

****PLAYING:**** When your client initiates playback of an audio stream, AudioPlayer must transition from an *idle*state to *playing*.

If you receive a directive instructing your client to perform an action, such as pausing or stopping the audio stream, if the client has trouble buffering the stream, or if playback fails, AudioPlayer must transition to the appropriate state when the action is performed (and send an event to TVS). Otherwise, AudioPlayer must remain in the *playing* state until the current stream has finished.

Additionally, AudioPlayer must remain in the playing state when:

* Reporting playback progress to TVS
* Sending stream metadata to TVS

****STOPPED:**** There are four instances when AudioPlayer must transition to the *stopped* state. While in the playing state, AudioPlayer must transition to *stopped* when:

* An issue with the stream is encountered and playback fails
* The client receives a [Stop](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "stop) directive from TVS
* A [ClearQueue](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "clearqueue) directive with a clearBehavior of CLEAR\_ALL is received
* A [Play](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "play) directive with a playBehavior of REPLACE\_ALL is received

While in the *paused* or *buffer\_underrun* states, AudioPlayer must transition to *stopped* when a ClearQueuedirective to CLEAR\_ALL is received.

AudioPlayer must transition from *stopped* to *playing* whenever your client receives a Play directive, starts playing an audio stream, and sends a [PlaybackStarted](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackstarted) event to the TVS.

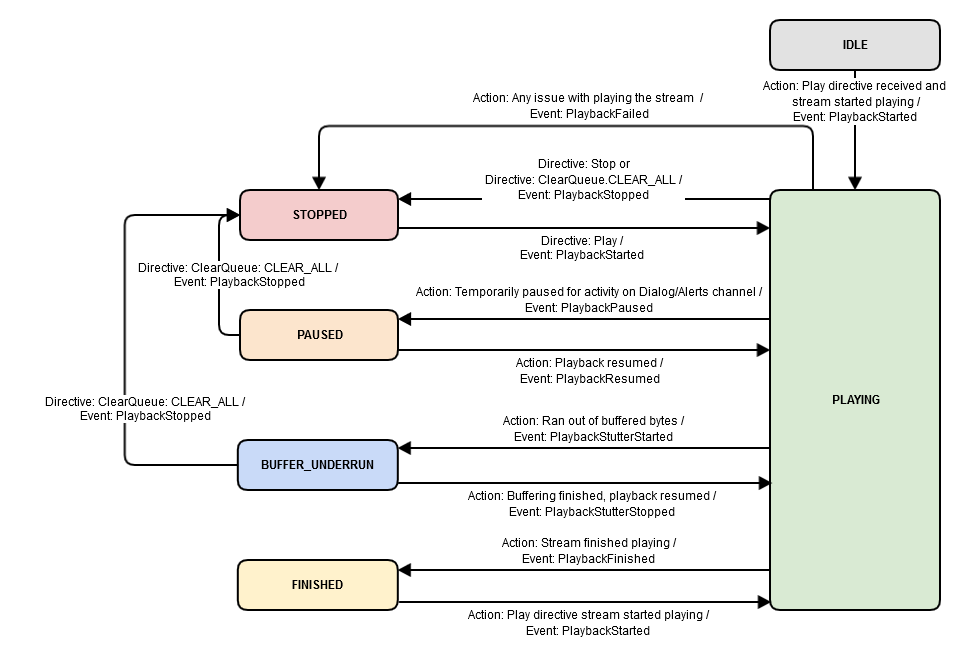
****PAUSED:**** AudioPlayer must transition to the *paused* state when audio on the Content channel is paused to accommodate a higher priority input/output (such as user orTVS speech). Playback must resume when the prioritized activity completes. For more information on prioritizing audio input/outputs, see [Interaction Model](https://developer.amazon.com/docs/alexa-voice-service/interaction-model.html).

****BUFFER\_UNDERRUN:**** AudioPlayer must transition to the *buffer\_underrun* state when the client is being fed data slower than it is being read. AudioPlayer must remain in this state until the buffer is full enough to resume playback, at which point it must return to the *playing* state.

****FINISHED:**** When a stream is finished playing, AudioPlayer must transition to the *finished* state. This is true for every stream in your playback queue. Even if there are streams queued to play, your client is required to send a [PlaybackFinished](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackfinished) event to TVS, and subsequently, transition from the *playing* state to finished when each stream is finished playing.

AudioPlayer must transition from *finished* to *playing* when:

* The client receives a [Play](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "play) directive
* The next stream in the playback queue starts playing (following a [PlaybackStarted](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackstarted) event).



### **AudioPlayer Context**

TVS expects a client to report playerActivity (state), and the offsetInMilliseconds for the currently playing media item with each event that requires context.

****Sample Message****

{

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackState"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}},

"playerActivity": "{{STRING}}"

}

}

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | This must match the token for the currently playing media item. Otherwise, the token must match what was provided in the last Play directive received. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |
| playerActivity | Identifies the component state of AudioPlayer.  ****Accepted Values****: IDLE, PLAYING, STOPPED, PAUSED, BUFFER\_UNDERRUN, and FINISHED. | string |

|  |  |
| --- | --- |
| ****Player Activity**** | ****Description**** |
| IDLE | Nothing was playing, no enqueued items. |
| PLAYING | Stream was playing. |
| PAUSED | Stream was paused. |
| BUFFER\_UNDERRUN | Buffer underrun. |
| FINISHED | Stream was finished playing. |
| STOPPED | Stream was interrupted. |

### **Play Directive**

The Play directive is sent to your client to initiate audio playback. It is a multipart message comprised of a JSON directive, and up to one audio stream or binary audio attachment.

**Note:** Learn more about **[Binary Audio Attachments](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "binary)**.

The playBehavior parameter included in the directive's payload can be used to determine how a client must handle queueing and playback of a stream. The accepted values provide hints for what action must be taken:

* ****REPLACE\_ALL:**** Immediately begin playback of the stream returned with the Play directive, and replace current and enqueued streams. When a stream is playing and you receive a Play directive with a playBehavior of REPLACE\_ALL you must send a [PlaybackStopped](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackstopped) event to TVS.
* ****ENQUEUE:**** Adds a stream to the end of the current queue.
* ****REPLACE\_ENQUEUED:**** Replace all streams in the queue. This does not impact the currently playing stream.

**Note:** When adding streams to your playback queue, you must ensure that the token for the currently playing stream matches the expectedPreviousToken in the stream being added to the queue. If the tokens do not match the stream must be ignored. However, if no expectedPreviousToken is returned, the stream must be added to the queue.

****Sample Message****

{

"directive": {

"header": {

"namespace": "AudioPlayer",

"name": "Play",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"playBehavior": "{{STRING}}",

"audioItem": {

"audioItemId": "{{STRING}}",

"stream": {

"url": "{{STRING}}",

"streamFormat": "AUDIO\_MPEG"

"offsetInMilliseconds": {{LONG}},

"expiryTime": "{{STRING}}",

"progressReport": {

"progressReportDelayInMilliseconds": {{LONG}},

"progressReportIntervalInMilliseconds": {{LONG}}

},

"token": "{{STRING}}",

"expectedPreviousToken": "{{STRING}}"

}

}

}

}

}

****Binary Audio Attachment****

Play directives may have a corresponding binary audio attachment as one part of the multipart message. When a binary audio attachment is present, the value provided for url will include the following prefix: cid.

The following multipart headers will precede the binary audio attachment:

Content-Type: application/octet-stream

Content-ID: {{Audio Item CID}}

{{BINARY AUDIO ATTACHMENT}}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

****Payload Parameters****

**Important:** Your client must be designed to accommodate and act on all payload parameters supported by Play, and must not break if key/value pairs in the JSON are missing.

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Type |
| playBehavior | Provides playback hints. Accepted values: REPLACE\_ALL, ENQUEUE, and REPLACE\_ENQUEUED.   * ****REPLACE\_ALL****: Immediately begin playback of the stream returned with the Play directive, and replace current and enqueued streams. * ****ENQUEUE****: Adds a stream to the end of the current queue. * ****REPLACE\_ENQUEUED****: Replace all streams in the queue. This does not impact the currently playing stream. | string |
| audioItem | Contains key/value pairs for audioItems. | object |
| audioItem.audioItemId | Identifies the audioItem. | string |
| audioItem.stream | Contains key/value pairs for streams. | object |
| audioItem.stream.url | Identifies the location of audio content. If the audio content is a binary audio attachment, the value will be a unique identifier for the content, which is formatted as follows: "cid:". Otherwise the value will be a remote http/https location. | string |
| audioItem.stream.streamFormat | streamFormat is included in the payload when the Playdirective has an associated binary audio attachment. This parameter ****will not**** appear if the associated audio is a stream.  ****Accepted Value****: AUDIO\_MPEG | string |
| audioItem.stream.offsetInMilliseconds | A timestamp indicating where in the stream the client must start playback. For example, when offsetInMilliseconds is set to 0, this indicates playback of the stream must start at 0, or the start of the stream. Any other value indicates that playback must start from the provided offset. | long |
| audioItem.stream.expiryTime | The date and time in ISO 8601 format for when the stream becomes invalid. | string |
| audioItem.stream.progressReport | Contains key/value pairs for progress reports. | object |
| audioItem.stream.progressReport. progressReportDelayInMilliseconds | Specifies (in milliseconds) when to send the [ProgressReportDelayElapsed](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "progressreportdelayelapsed) event to TVS. ProgressReportDelayElapsed must only be sent once at the specified interval. ****Please note****: Some music providers do not require this report. If the report is not required, progressReportDelayInMilliseconds will not appear in the payload. | long |
| audioItem.stream.progressReport. progressReportIntervalInMilliseconds | Specifies (in milliseconds) when to emit a [ProgressReportIntervalElapsed](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "progressreportintervalelapsed) event to TVS. ProgressReportIntervalElapsed must be sent periodically at the specified interval. ****Please note****: Some music providers do not require this report. If the report is not required, progressReportIntervalInMilliseconds will not appear in the payload. | long |
| audioItem.stream.token | An opaque token that represents the current stream. | string |
| audioItem.stream. expectedPreviousToken | An opaque token that represents the expected previous stream. | string |

### **PlaybackStarted Event**

The PlaybackStarted event must be sent to TVS after your client processes a [Play](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "play) directive and begins playback of the associated audio stream.

**Note:** For each URL that TVS sends, it expects no more than one PlaybackStarted event. If you receive a playlist URL (composed of multiple URLs) ****only**** send one PlaybackStarted event

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackStarted",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |

### **PlaybackNearlyFinished Event**

The PlaybackNearlyFinished event must be sent when your client is ready to buffer/download the next stream in your playback queue. Your client must ensure that this event is only sent following a [PlaybackStarted](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackstarted) event for the currently playing stream. TVS will respond to this event with one of the following:

* A [Play](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "play) directive containing the next stream
* An HTTP 204 response code

**Tip:** As a best practice, you may want to consider waiting until the previous song has buffered before sending a PlaybackNearlyFinished event to TVS. This lowers the risk of exceeding the expiryTime and can reduce the frequency of playback stutters that may occur when downloading and processing multiple Play directives at the same time.

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackNearlyFinished",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |

### **ProgressReportDelayElapsed Event**

The ProgressReportDelayElapsed event must be sent to TVS if progressReportDelayInMilliseconds is present in the [Play](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "play) directive. The event must be sent once at the specified interval from the start of the stream (not from the offsetInMilliseconds). For example, if the Play directive contains progressReportDelayInMilliseconds with a value of 20000, the ProgressReportDelayElapsed event must be sent 20,000 milliseconds from the start of the track. However, if the Play directive contains an offsetInMilliseconds value of 10000 and progressReportDelayInMilliseconds value 20000, the event must be sent 10,000 milliseconds into playback. This is because the progress report is sent from the start of a stream, not the Play directive's offset.

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "ProgressReportDelayElapsed",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |

### **ProgressReportIntervalElapsed Event**

The ProgressReportIntervalElapsed event must be sent to TVS if progressReportIntervalInMilliseconds is present in the [Play](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "play) directive. The event must be sent periodically at the specified interval from the start of the stream (not from the offsetInMilliseconds). For example, if the Play directive contains progressReportIntervalInMilliseconds with a value of 20000, the ProgressReportIntervalElapsed event must be sent 20,000 milliseconds from the start of the track, and every 20,000 milliseconds until the stream ends. However, if the Play directive contains an offsetInMilliseconds value of 10000 and a progressReportIntervalInMilliseconds value of 20000, the event must be sent 10,000 milliseconds from the start of playback, and every 20,000 milliseconds after that until the stream ends. This is because the interval specified is from the start of the stream, not the Play directive's offset.

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "ProgressReportIntervalElapsed",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |

### **PlaybackStutterStarted Event**

The PlaybackStutterStarted event must be sent to TVS, following a [PlaybackStarted](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackstarted) event, when the client's AudioPlayer component is being fed data slower than it is being read. The component must transition to the *buffer\_underrun* state once this event has been sent and remain in this state until the buffer is full enough to resume playback.

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackStutterStarted",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

| ****Parameter**** | ****Description**** | ****Type**** |
| --- | --- | --- |
| token | An opaque token provided by the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |

### **PlaybackStutterFinished Event**

The PlaybackStutterFinished event must be sent to TVS when the buffer is full enough to resume playback of a stream. TVS doesn't expect a subsequent [PlaybackStarted](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackstarted) event when audio playback resumes.

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackStutterFinished",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}},

"stutterDurationInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |
| stutterDurationInMilliseconds | Identifies the duration of a stutter in milliseconds. | long |

### **PlaybackFinished Event**

The PlaybackFinished event must be sent to TVS when your client finishes playback of a stream.

This event ****is not**** sent when:

* Playback is stopped (either locally or as the result of a [Stop](https://developer.amazon.com/docs/alexa-voice-service/audioplayer.html" \l "stop) directive)
* Navigating between streams (next/previous)

**Note:** For each URL that TVS sends, it expects no more than one PlaybackFinished event. If you receive a playlist URL (composed of multiple URLs) ****only**** send one PlaybackFinished event

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackFinished",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |

### **PlaybackFailed Event**

The PlaybackFailed event must be sent to TVS whenever your client encounters an error while attempting to play a stream. It is possible for the currentPlaybackToken to be different from the token in the payload in cases where a stream is playing and the next stream fails to buffer.

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackFailed",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"currentPlaybackState": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}},

"playerActivity": "{{STRING}}"

},

"error": {

"type": "{{STRING}}",

"message": "{{STRING}}"

}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the Play directive that represents the stream that failed to playback. | string |
| currentPlaybackState | Contains key/value pairs for the playbackState object. | object |
| playbackState.token | An opaque token provided by the Play directive. | string |
| playbackState.offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |
| playbackState.playerActivity | Identifies the player state.  ****Accepted values****: PLAYING, STOPPED, PAUSED, FINISHED, BUFFER\_UNDERRUN, or IDLE. | string |
| error | Contains key/value pairs for error messages. | object |
| error.type | Identifies the specific type of error. The table below provides details for each error type. | string |
| error.message | A description of the error the device has encountered. This is used for logging purposes only. For HTTP related errors, the error message should contain the HTTP error response body if present. | string |

****Error Types****

|  |  |
| --- | --- |
| ****Value**** | ****Description**** |
| MEDIA\_ERROR\_UNKNOWN | An unknown error occurred. |
| MEDIA\_ERROR\_INVALID\_REQUEST | The server recognized the request as being malformed.  E.g. bad request, unauthorized, forbidden, not found, etc. |
| MEDIA\_ERROR\_SERVICE\_UNAVAILABLE | The client was unable to reach the service. |
| MEDIA\_ERROR\_INTERNAL\_SERVER\_ERROR | The server accepted the request, but was unable to process the request as expected. |
| MEDIA\_ERROR\_INTERNAL\_DEVICE\_ERROR | There was an internal error on the client. |

### **Stop Directive**

The Stop directive is sent to your client to stop playback of an audio stream. Your client may receive a Stopdirective as the result of a voice request, a physical button press or GUI affordance.

****Sample Message****

{

"directive": {

"header": {

"namespace": "AudioPlayer",

"name": "Stop",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

### **PlaybackStopped Event**

The PlaybackStopped event must be sent to TVS when your client receives one of the following directives and stops playback of an audio stream:

* A [Stop](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "stop) directive
* A [Play](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "play) directive with a playBehavior of REPLACE\_ALL
* A [ClearQueue](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "clearqueue) directive with a clearBehavior of CLEAR\_ALL

**Note:** This event is only sent when a stream is terminated as a result of receiving one of the directives listed above. Typically, this is the result of a user action. This event ****must not**** be sent when a stream has finished playing (see [PlaybackFinished](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackfinished)).

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackStopped",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |

### **PlaybackPaused Event**

The PlaybackPaused event must be sent when your client temporarily pauses audio on the Content channel to accommodate a higher priority input/output. Playback must resume when the prioritized activity completes; at which point your client must send a [PlaybackResumed](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackresumed) event. For more information on prioritizing audio input/outputs, see [Interaction Model](https://developer.amazon.com/docs/alexa-voice-service/interaction-model.html).

**Note:** PlaybackPaused should be sent after a Recognize event to reduce latency.

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackPaused",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided in the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |

### **PlaybackResumed Event**

The PlaybackResumed event must be sent to TVS when playback resumes following a [PlaybackPaused](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "playbackpaused) event (when playback is temporarily paused on the Content channel to accommodate a higher priority input/output). For more information on prioritizing audio input/outputs, see [Interaction Model](https://developer.amazon.com/docs/alexa-voice-service/interaction-model.html).

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackResumed",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided in the Play directive. | string |
| offsetInMilliseconds | Identifies a track's current offset in milliseconds. The value sent must be equal to or greater than zero. Negative values are not accepted. | long |

### **ClearQueue Directive**

The ClearQueue directive is sent from TVS to your client to clear the playback queue. The ClearQueuedirective has two behaviors: CLEAR\_ENQUEUED, which clears the queue and continues to play the currently playing stream; and CLEAR\_ALL, which clears the entire playback queue and stops the currently playing stream (if applicable).

****Sample Message****

{

"directive": {

"header": {

"namespace": "AudioPlayer",

"name": "ClearQueue",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"clearBehavior": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| clearBehavior | A string value used to determine clear queue behavior.  ****Accepted values****: CLEAR\_ENQUEUED and CLEAR\_ALL | string |

### **PlaybackQueueCleared Event**

The PlaybackQueueCleared event must be sent to TVS after your client handles a [ClearQueue](https://developer.amazon.com/zh/docs/alexa-voice-service/audioplayer.html" \l "clearqueue) directive.

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "PlaybackQueueCleared",

"messageId": "{{STRING}}"

},

"payload": {

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

An empty payload must be sent.

### **StreamMetadataExtracted Event**

If metadata is available for an audio stream that your client receives and starts playing: your client should take the key/value pairs received as raw data and translate those pairs into a JSON object. In this JSON object, strings and numbers should be represented as JSON strings, and booleans should be represented as JSON booleans. Your client should filter out any tags containing binary data. For example, your client should not send the image, image preview, attachment, or application data tags to TVS.

****Sample Message****

{

"event": {

"header": {

"namespace": "AudioPlayer",

"name": "StreamMetadataExtracted",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}",

"metadata": {

"{{STRING}}": "{{STRING}}",

"{{STRING}}": {{BOOLEAN}}

"{{STRING}}": "{{STRING NUMBER}}"

}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided by the Play directive. | string |
| metadata | Contains key/value pairs associated with the metadata received. | object |

## Notifications interface

The Notifications interface allows TVS to inform users that new content is available from TVS domains or an enabled TVS skill. Specifically, the interface exposes two directives that instruct your client to render and clear visual and audio indicators for the user.

This interface ****does not**** provide the content for a notification, it only provides the audio and visual indicators that are used to inform the user that new content is available. For example, the product may flash a yellow LED and play an audio file, at which time a user can retrieve any pending notifications by asking, "DingDangDingDang, what did I miss?" or "DingDangDingDang, what are my notifications?"

For information about flow and delivery, the do not disturb setting, and UX considerations, see the [Notifications Overview](https://developer.amazon.com/docs/alexa-voice-service/notifications-overview.html).

### **Notifications Context**

TVS expects a client to report the state of a product's notification indicator with each event that requires context.

To learn more about reporting Context, see [Context Overview](https://developer.amazon.com/zh/docs/alexa-voice-service/context.html).

****Sample Message****

{

"header": {

"namespace": "Notifications",

"name": "IndicatorState"

},

"payload": {

"isEnabled": {{BOOLEAN}},

"isVisualIndicatorPersisted": {{BOOLEAN}}

}

}

****Payload Parameters****

| ****Parameter**** | ****Description**** | ****Type**** |
| --- | --- | --- |
| isEnabled | Indicates there are new or pending notifications that have not been communicated to the user.  ****Note****: Any indicator that has not been cleared is considered enabled. | boolean |
| isVisualIndicatorPersisted | Corresponds to the persistVisualIndicator value of the last SetIndicator directive received. If persistVisualIndicator was true for the last directive received, upon reconnecting, isVisualIndicatorPersisted must be true. | boolean |

### **SetIndicator Directive**

This directive instructs your client to render visual and audio indicators when a notification is available to be retrieved. Your client may receive multiple SetIndicator directives in a short period of time. If directives overlap, consider these rules:

* If the assetId of the current directive matches the assetId of the incoming directive, ****DO NOT**** play the asset.
* If the assetId of the current directive does not match the assetId of the incoming directive, play the asset for the incoming directive ****AFTER**** playback of the current asset is finished.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Notifications",

"name": "SetIndicator",

"messageId": "{{STRING}}"

},

"payload": {

"persistVisualIndicator": {{BOOLEAN}},

"playAudioIndicator": {{BOOLEAN}},

"asset": {

"assetId": "{{STRING}}",

"url": "{{STRING}}"

}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Type |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Type |
| persistVisualIndicator | Specifies if your product must display a persistent visual indicator (if applicable) after handling this directive. | boolean |
| playAudioIndicator | Specifies if your product must play an audio indicator when this directive is handled. | boolean |
| asset | Contains information about the audio asset that must be played if playAudioIndicator is true. | object |
| asset.assetId | A unique identifier for the asset. | string |
| asset.url | This asset may be downloaded and cached by your client. The URL provided is valid for 60 minutes. If the product is offline, or if the asset isn’t available, your product should play the default indicator sound. | string |

### **ClearIndicator Directive**

This directive instructs your client to clear all active visual and audio indicators.

* If an audio indicator is playing when this directive is received, it should be stopped immediately.
* If any visual indicators are set when this directive is received, they should be cleared immediately.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Notifications",

"name": "ClearIndicator",

"messageId": "{{STRING}}"

},

"payload": {

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Type |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

This directive has an empty payload.

## PlaybackController Interface

The PlaybackController Interface exposes a series of events for navigating a playback queue with an on-client button press or GUI affordance, rather than through a speech request.

### **PlayCommandIssued Event**

The PlayCommandIssued event must be sent when a user starts/resumes playback of a media item using an on-client button press or GUI affordance.

****Sample Message****

{

"context": [

// This is an array of context objects that are used to communicate the

// state of all client components to TVS. See Context for details.

],

"event": {

"header": {

"namespace": "PlaybackController",

"name": "PlayCommandIssued",

"messageId": "{{STRING}}"

},

"payload": {

}

}

}

****Context****

This event requires the client to send the status of all client component states to TVS. For additional information see [Context](https://developer.amazon.com/docs/alexa-voice-service/context.html).

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

An empty payload should be sent.

### **PauseCommandIssued Event**

The PauseCommandIssued event must be sent when a user pauses the playback of a media item using an on-client button press or GUI affordance.

****Sample Message****

{

"context": [

// This is an array of context objects that are used to communicate the

// state of all client components to TVS. See Context for details.

],

"event": {

"header": {

"namespace": "PlaybackController",

"name": "PauseCommandIssued",

"messageId": "{{STRING}}"

},

"payload": {

}

}

}

****Context****

This event requires the client to send the status of all client component states to TVS. For additional information see [Context](https://developer.amazon.com/docs/alexa-voice-service/context.html).

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

An empty payload should be sent.

### **NextCommandIssued Event**

The NextCommandIssued event must be sent when a user skips to the next media item in their playback queue using an on-client button press or GUI affordance.

****Sample Message****

{

"context": [

// This is an array of context objects that are used to communicate the

// state of all client components to TVS. See Context for details.

],

"event": {

"header": {

"namespace": "PlaybackController",

"name": "NextCommandIssued",

"messageId": "{{STRING}}"

},

"payload": {

}

}

}

****Context****

This event requires the client to send the status of all client component states to TVS. For additional information see [Context](https://developer.amazon.com/docs/alexa-voice-service/context.html).

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

An empty payload should be sent.

### **PreviousCommandIssued Event**

The PreviousCommandIssued event must be sent when a user skips to the previous media item in their playback queue using an on-client button press or GUI affordance.

****Sample Message****

{

"context": [

// This is an array of context objects that are used to communicate the

// state of all client components to TVS. See Context for details.

],

"event": {

"header": {

"namespace": "PlaybackController",

"name": "PreviousCommandIssued",

"messageId": "{{STRING}}"

},

"payload": {

}

}

}

****Context****

This event requires the client to send the status of all client component states to TVS.

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

An empty payload should be sent.

### **ButtonCommandIssued Event**

This event is used to notify TVS of a unique on-client button press or GUI affordance, such as skip forward or skip backward. Skip duration is determined by the provider/skill, and each event is additive. For example, if a user presses the skip forward button three times in a row, and as a result three ButtonCommandIssued events are sent to TVS, the additive effect, if the skip is 30 seconds, will be 90 seconds.

****Sample Message****

{

"context": [

// This is an array of context objects that are used to communicate the

// state of all client components to TVS. See Context for details.

],

"event": {

"header": {

"namespace": "PlaybackController",

"name": "ButtonCommandIssued",

"messageId": "{{STRING}}"

},

"payload": {

"name": "{{STRING}}"

}

}

}

****Context****

This event requires the client to send the status of all client component states to TVS.

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| name | Specifies the command triggered by an on-client button press or GUI affordance. ****Accepted values****: SKIPFORWARD, SKIPBACKWARD. | string |

### **ToggleCommandIssued Event**

This event is used to notify TVS that an option or feature has been selected or deselected using an on-client button press or GUI affordance. Supported options include: shuffle, loop, repeat, thumbs up, and thumbs down.

****Sample Message****

{

"context": [

// This is an array of context objects that are used to communicate the

// state of all client components to TVS. See Context for details.

],

"event": {

"header": {

"namespace": "PlaybackController",

"name": "ToggleCommandIssued",

"messageId": "{{STRING}}"

},

"payload": {

"name": "{{STRING}}"

"action": "{{STRING}}"

}

}

}

****Context****

This event requires the client to send the status of all client component states to TVS.

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| name | Specifies the option/feature that is being toggled by an on-client button press or GUI affordance. ****Accepted values****: SHUFFLE, LOOP, REPEAT, THUMBSUP, THUMBSDOWN. | string |
| action | Indicates if the toggle has been selected or deselected. ****Accepted values:**** SELECT, DESELECT. | string |

## Settings interface

The Settings interface is used to manage TVS settings on your product, such as locale.

### **SettingsUpdated Event**

The SettingsUpdated event must be sent when TVS settings are adjusted using on-product controls or a companion app. For example, your user may change their locale setting from US (en-US) to Germany (de-DE) using your companion app. When this happens, your client must notify TVS of the change with the SettingsUpdated event.

**Note:** If a malformed or unsupported value is sent to TVS an exception message is returned.

****Sample Message****

{

"event": {

"header": {

"namespace": "Settings",

"name": "SettingsUpdated",

"messageId": "{{STRING}}"

},

"payload": {

"settings": [

{

"key": "{{STRING}}",

"value": "{{STRING}}"

}

]

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Type |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Type |
| settings | A list of TVS settings for a product. Each item in the list is an object with key/value pairs. | list |
| key | Identifies the key. Use the table below to identify accepted keys. | string |
| value | Identifies the key's value. Use the table below to identify accepted values for each key. | string |

****Accepted Key/Value Pairs****

|  |  |
| --- | --- |
| Accepted Keys | Accepted Values |
| locale | de-DE, en-AU, en-CA, en-GB, en-IN, en-US, fr-FR, ja-JP |

## Speaker interface

The Speaker interface exposes directives and events that are used to adjust volume and mute/unmute a client's speaker. TVS supports two methods for volume adjustment, which are exposed through the [SetVolume](https://developer.amazon.com/zh/docs/alexa-voice-service/speaker.html" \l "setvolume) and [AdjustVolume](https://developer.amazon.com/zh/docs/alexa-voice-service/speaker.html" \l "adjustvolume) directives.

### **Speaker Context**

TVS expects a client to report volume and muted state information for the Speaker interface with each event that requires context.

To learn more about reporting Context, see [Context Overview](https://developer.amazon.com/zh/docs/alexa-voice-service/context.html).

****Sample Message****

{

"header": {

"namespace": "Speaker",

"name": "VolumeState"

},

"payload": {

"volume": {{LONG}},

"muted": {{BOOLEAN}}

}

}

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| volume | Identifies current speaker volume.  ****Accepted Values****: 0 to 100 | long |
| muted | Identifies mute state of the client's speaker. | boolean |

### **SetVolume Directive**

This directive instructs your client to make an absolute volume adjustment. The volume value will be between ****0**** (min) and ****100**** (max), inclusive.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Speaker",

"name": "SetVolume",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"volume": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| volume | The absolute volume level scaled from 0 (min) to 100 (max).  ****Accepted values:**** Any value between ****0**** and ****100****. | long |

### **AdjustVolume Directive**

This directive instructs your client to make a relative volume adjustment. The volume value will be between ****-100**** and ****100****, inclusive.

The AdjustVolume directive is always relative to the current volume setting and is positive to increase volume, or negative to reduce volume.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Speaker",

"name": "AdjustVolume",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"volume": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| volume | The relative volume adjustment. A positive or negative long value used to increase or decrease volume in relation to the current volume setting.  ****Accepted values:**** Any value between ****-100**** and ****100****, inclusive. | long |

### **VolumeChanged Event**

The VolumeChanged event must be sent to TVS when:

* A [SetVolume](https://developer.amazon.com/zh/docs/alexa-voice-service/speaker.html" \l "setvolume) or [AdjustVolume](https://developer.amazon.com/zh/docs/alexa-voice-service/speaker.html" \l "adjustvolume) directive is received and processed to indicate that the speaker volume on your product has been adjusted/changed.
* Volume is locally adjusted to indicate that the speaker volume on your product has been adjusted/changed.  
  ****Important****: volume must be a value between ****0**** (min) and ****100**** (max), inclusive. If your product locally supports volume adjustment from 0 to 10, when the user increases the volume to 8, TVS expects the volume value sent to be 80.

****Sample Message****

{

"event": {

"header": {

"namespace": "Speaker",

"name": "VolumeChanged",

"messageId": "{{STRING}}"

},

"payload": {

"volume": {{LONG}},

"muted": {{BOOLEAN}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| volume | The absolute volume level scaled from 0 (min) to 100 (max).  ****Accepted values:**** Any long value between ****0**** and ****100**** | long |
| mute | A boolean value is used to mute/unmute a product's speaker. The value is true when the speaker is muted, and false when unmuted. | boolean |

### **SetMute Directive**

This directive is sent from TVS to your client to mute the product's speaker.

****Sample Message****

{

"directive": {

"header": {

"namespace": "Speaker",

"name": "SetMute",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"mute": {{BOOLEAN}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| mute | A boolean value is used to mute/unmute a product's speaker. The value is true when the speaker is muted, and false when unmuted. | boolean |

### **MuteChanged Event**

The MuteChanged event must be sent to TVS when:

* A [SetMute](https://developer.amazon.com/zh/docs/alexa-voice-service/speaker.html" \l "setmute) directive is received and processed to indicate that the mute status of the product's speaker has changed.
* Your product is muted/unmuted locally to indicate that the mute status of the product's speaker has changed.

****Sample Message****

{

"event": {

"header": {

"namespace": "Speaker",

"name": "MuteChanged",

"messageId": "{{STRING}}"

},

"payload": {

"volume": {{LONG}},

"muted": {{BOOLEAN}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| volume | The absolute volume level scaled from 0 (min) to 100 (max).  ****Accepted values:**** Any long value between ****0**** and ****100**** | long |
| mute | A boolean value is used to mute/unmute a product's speaker. The value is true when the speaker is muted, and false when unmuted. | boolean |

## SpeechRecognizer interface

Every user utterance leverages SpeechRecognizer. It is the core interface of the Tencent Voice Service (TVS). It exposes directives and events for capturing user speech and prompting a client when TVS needs additional speech input.

Additionally, this interface allows your client to inform TVS of how an interaction with TVS was initiated ([press and hold, tap and release, voice-initiated/wake word enabled](https://developer.amazon.com/docs/alexa-voice-service/audio-hardware-configurations.html" \l "applications)), and choose the appropriate [Automatic Speech Recognition (ASR) profile](https://developer.amazon.com/docs/alexa-voice-service/audio-hardware-configurations.html" \l "asr) for your product, which allows TVS to understand user speech and respond with precision.

### **State Diagram**

The following diagram illustrates state changes driven by SpeechRecognizer components. Boxes represent SpeechRecognizer states and the connectors indicate state transitions.

SpeechRecognizer has the following states:

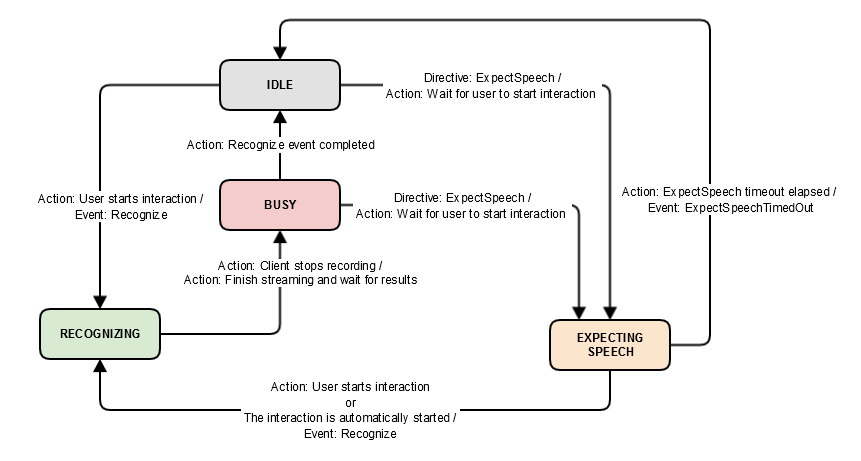
****IDLE:**** Prior to capturing user speech, SpeechRecognizer should be in an *idle* state. SpeechRecognizer should also return to an *idle* state after a speech interaction with TVS has concluded. This can occur when a speech request has been successfully processed or when an [ExpectSpeechTimedOut](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "expectspeechtimedout) event has elapsed.

Additionally, SpeechRecognizer may return to an *idle* state during a multiturn interaction, at which point, if additional speech is required from the user, it should transition from the *idle* state to the *expecting speech*state without a user starting a new interaction.

****RECOGNIZING:**** When a user begins interacting with your client, specifically when captured audio is streamed to TVS, SpeechRecognizer should transition from the *idle* state to the *recognizing state*. It should remain in the *recognizing state* until the client stops recording speech (or streaming is complete), at which point your SpeechRecognizer component should transition from the *recognizing* state to the *busy* state.

****BUSY:**** While processing the speech request, SpeechRecognizer should be in the *busy* state. You cannot start another speech request until the component transitions out of the *busy* state. From the *busy* state, SpeechRecognizer will transition to the *idle* state if the request is successfully processed (completed) or to the *expecting speech* state if TVS requires additional speech input from the user.

****EXPECTING SPEECH:**** SpeechRecognizer should be in the *expecting speech* state when additional audio input is required from a user. From *expecting speech*, SpeechRecognizer should transition to the *recognizing state*when a user interaction occurs or the interaction is automatically started on the user's behalf. It should transition to the *idle* state if no user interaction is detected within the specified timeout window.



### **SpeechRecognizer Context**

TVS expects all clients to report the currently set wake word, if wake word enabled.

****Sample Message****

{

"header": {

"namespace": "SpeechRecognizer",

"name": "RecognizerState"

},

"payload": {

"wakeword": "DING1DANG1DING1DANG"

}

}

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| wakeword | Identifies the current wake word.  ****Accepted Value****: "DING1DANG1DING1DANG" | string |

### **Recognize Event**

The Recognize event is used to send user speech to TVS and translate that speech into one or more directives. This event must be sent as a multipart message: the first part a JSON-formatted object, the second, binary audio captured by the product's microphone. We encourage streaming (chunking) captured audio to the Tencent Voice Service to reduce latency; the stream should contain 10ms of captured audio per chunk (320 bytes).

After an interaction with TVS is initiated, the microphone must remain open until:

* A StopCapture directive is received.
* The stream is closed by the TVS service.
* The user manually closes the microphone. For example, a [press and hold implementation](https://developer.amazon.com/docs/alexa-voice-service/audio-hardware-configurations.html" \l "applications).

The profile parameter and initiator object tell TVS which ASR profile should be used to best understand the captured audio being sent, and how the interaction with TVS was initiated.

All captured audio sent to TVS should be encoded as:

* 16bit Linear PCM
* 16kHz sample rate
* Single channel
* Little endian byte order

****Sample Message****

{

"context": [

// This is an array of context objects that are used to communicate the

// state of all client components to TVS. See Context for details.

],

"event": {

"header": {

"namespace": "SpeechRecognizer",

"name": "Recognize",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"profile": "{{STRING}}",

"format": "{{STRING}}",

"initiator": {

"type": "{{STRING}}",

"payload": {

"wakeWordIndices": {

"startIndexInSamples": {{LONG}},

"endIndexInSamples": {{LONG}}

},

"token": "{{STRING}}"

}

}

}

}

}

****Binary Audio Attachment****

Each Recognize event requires a corresponding binary audio attachment as one part of the multipart message. The following headers are required for each binary audio attachment:

Content-Disposition: form-data; name="audio"

Content-Type: application/octet-stream

{{BINARY AUDIO ATTACHMENT}}

****Context****

This event requires your product to report the status of all client component states to TVS in the context object.

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique identifier that your client must create for each Recognizeevent sent to TVS. This parameter is used to correlate directives sent in response to a specific Recognize event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| [profile](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "profile) | Identifies the Automatic Speech Recognition (ASR) profile associated with your product. TVS supports three distinct ASR profiles optimized for user speech from varying distances.  ****Accepted values:**** CLOSE\_TALK, NEAR\_FIELD, FAR\_FIELD. | string |
| format | Identifies the format of captured audio.  ****Accepted value:**** AUDIO\_L16\_RATE\_16000\_CHANNELS\_1. | string |
| [initiator](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "initiator) | Lets TVS know how an interaction was initiated.  This object is required when an interaction is originated by the end user (wake word, tap, push and hold).  If initiator is present in an ExpectSpeech directive then it must be returned in the following Recognizeevent. If initiator is absent from the ExpectSpeech directive, then it should ****not**** be included in the following Recognize event. | object |
| initiator.type | Represents the action taken by a user to initiate an interaction with TVS.  ****Accepted values:**** PRESS\_AND\_HOLD, TAP, and WAKEWORD. If an initiator.type is provided in an ExpectSpeech directive, that string must be returned as initiator.type in the following Recognize event. | string |
| initiator.payload | Includes information about the initiator. | object |
| initiator.payload.wakeWordIndices | This object is required when initiator.type is set to WAKEWORD.  wakeWordIndices includes the startIndexInSamples and endIndexInSamples. For additional details, see [Requirements for Cloud-Based Wake Word Verification](https://developer.amazon.com/docs/alexa-voice-service/streaming-requirements-for-cloud-based-wake-word-verification.html). | object |
| initiator.payload.wakeWordIndices.startIndexInSamples | Represents the index in the audio stream where the wake word starts (in samples). The start index should be accurate to within 50ms of wake word detection. | long |
| initiator.payload.wakeWordIndices.endIndexInSamples | Represents the index in the audio stream where the wake word ends (in samples). The end index should be accurate to within 150ms of the end of the detected wake word. | long |
| initiator.payload.token | An opaque string. This value is only required if present in the payload of a preceding [ExpectSpeech](https://developer.amazon.com/docs/alexa-voice-service/speechrecognizer.html" \l "expectspeech) directive. | string |

****Profiles****

ASR profiles are tuned for different products, form factors, acoustic environments and use cases. Use the table below to learn more about accepted values for the profile parameter.

|  |  |
| --- | --- |
| Value | Optimal Listening Distance |
| CLOSE\_TALK | 0 to 2.5 ft. |
| NEAR\_FIELD | 0 to 5 ft. |
| FAR\_FIELD | 0 to 20+ ft. |

****Initiator****

The initiator parameter tells TVS how an interaction with TVS was triggered; and determines two things:

1. If StopCapture will be sent to your client when the end of speech is detected in the cloud.
2. If cloud-based wake word verification will be performed on the stream.

initiator must be included in the payload of each SpeechRecognizer.Recognize event. The following values are accepted:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Value | Description | Supported Profile(s) | StopCapture Enabled | Wake Word Verification Enabled | Wake Word Indices Required |
| PRESS\_AND\_HOLD | Audio stream initiated by pressing a button (physical or GUI) and terminated by releasing it. | CLOSE\_TALK | N | N | N |
| TAP | Audio stream initiated by the tap and release of a button (physical or GUI) and terminated when a StopCapture   directive is received. | NEAR\_FIELD, FAR\_FIELD | Y | N | N |
| WAKEWORD | Audio stream initiated by the use of a wake word and terminated when a StopCapture   directive is received. | NEAR\_FIELD, FAR\_FIELD | Y | Y | Y |

### **StopCapture Directive**

This directive instructs your client to stop capturing a user’s speech after TVS has identified the user’s intent or when end of speech is detected. When this directive is received, your client must immediately close the microphone and stop listening for the user’s speech.

**Note:** StopCapture is sent to your client on the downchannel stream and may be received while speech is still being streamed to TVS. To receive the StopCapture directive, you must use a profile in your [Recognize](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "recognize) event that supports cloud-endpointing, such as NEAR\_FIELD or FAR\_FIELD.

****Sample Message****

{

"directive": {

"header": {

"namespace": "SpeechRecognizer",

"name": "StopCapture",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "recognize) event. | string |

### **ExpectSpeech Directive**

ExpectSpeech is sent when TVS requires additional information to fulfill a user's request. It instructs your client to open the microphone and begin streaming user speech. If the microphone is not opened within the specified timeout window, an [ExpectSpeechTimedOut](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "expectspeechtimedout) event must be sent from your client to TVS.

During a multi-turn interaction with TVS, your device will receive at least one ExpectSpeech directive instructing your client to start listening for user speech. If present, the initiator object included in the payload of the ExpectSpeech directive must be passed back to TVS as the initiator object in the following Recognize event. If initiator is absent from the payload, the following Recognize event should ****not**** include initiator.

****Sample Message****

{

"directive": {

"header": {

"namespace": "SpeechRecognizer",

"name": "ExpectSpeech",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"timeoutInMilliseconds": {{LONG}},

"initiator": {

"type": "{{STRING}}",

"payload": {

"token": "{{STRING}}"

}

}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| timeoutInMilliseconds | Specifies, in milliseconds, how long your client should wait for the microphone to open and begin streaming user speech to TVS. If the microphone is not opened within the specified timeout window, then the [ExpectSpeechTimedOut](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "expectspeechtimedout) event must be sent. The primary use case for this behavior is a PRESS\_AND\_HOLD implementation. | long |
| initiator | Contains information about the interaction. If present it must be sent back to TVS in the following [Recognize event](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "recognize). | object |
| initiator.type | An opaque string. If present it must be sent back to TVS in the following [Recognize event](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "recognize). | string |
| initiator.payload | Includes information about the initiator. | object |
| initiator.payload.token | An opaque string. If present it must be sent back to TVS in the following [Recognize event](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "recognize). | string |

### **ExpectSpeechTimedOut Event**

This event must be sent to TVS if an [ExpectSpeech](https://developer.amazon.com/zh/docs/alexa-voice-service/speechrecognizer.html" \l "expectspeech) directive was received, but was not satisfied within the specified timeout window.

****Sample Message****

{

"event": {

"header": {

"namespace": "SpeechRecognizer",

"name": "ExpectSpeechTimedOut",

"messageId": "{{STRING}}",

},

"payload": {

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

An empty payload should be sent.

## SpeechSynthesizer interface

When users ask your product a question or make a request, the SpeechSynthesizer interface is used to return TVS's speech response. For instance, when a user asks TVS, "What's the weather in Shanghai?" The TVS Voice Service will return a Speak directive to your client with a binary audio attachment, which your client should process and play. This page covers SpeechSynthesizer directives and events.

### **States**

SpeechSynthesizer has the following states:

****PLAYING:**** While TVS is speaking, SpeechSynthesizer should be in a playing state. SpeechSynthesizer should transition to the *finished* state when playback of TVS's speech is complete.

****FINISHED:**** When TVS is finished speaking, SpeechSynthesizer should transition to the *finished* state following a SpeechFinished event.

### **SpeechSynthesizer Context**

TVS expects a client to report playerActivity (state), and the offsetInMilliseconds for the currently playing TTS with each event that requires context.

****Sample Message****

{

"header": {

"namespace": "SpeechSynthesizer",

"name": "SpeechState"

},

"payload": {

"token": "{{STRING}}",

"offsetInMilliseconds": {{LONG}},

"playerActivity": "{{STRING}}"

}

}

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | An opaque token provided in the Speak directive. | string |
| offsetInMilliseconds | Identifies the current offset of TTS in milliseconds. | long |
| playerActivity | Identifies the component state of SpeechSynthesizer.  ****Accepted Values****: PLAYING or FINISHED | string |

|  |  |
| --- | --- |
| ****Player Activity**** | ****Description**** |
| PLAYING | Speech was playing. |
| FINISHED | Speech was finished playing. |

### **Speak Directive**

This directive is sent from TVS to your client any time a speech response from TVS is required. In most cases, the Speak directive is sent in response to a user request, such as a [Recognize](https://developer.amazon.com/docs/alexa-voice-service/speechrecognizer.html" \l "recognize) event. However, a Speak directive may also be sent to your client to preface an action that will be taken. For instance, when a user makes a request to set a timer, in addition to receiving a [SetAlert](https://developer.amazon.com/docs/alexa-voice-service/alerts.html" \l "setalert) directive that instructs the client to set an alarm, the client also receives a Speak directive which notifies the user that the timer was successfully set.

This directive is sent to your client as a multipart message: one part a JSON-formatted directive and one binary audio attachment.

****Sample Message****

{

"directive": {

"header": {

"namespace": "SpeechSynthesizer",

"name": "Speak",

"messageId": "{{STRING}}",

"dialogRequestId": "{{STRING}}"

},

"payload": {

"url": "{{STRING}}",

"format": "{{STRING}}",

"token": "{{STRING}}"

}

}

}

****Binary Audio Attachment****

Each Speak directive will have a corresponding binary audio attachment as one part of the multipart message. The following multipart headers will precede the binary audio attachment:

Content-Type: application/octet-stream

Content-ID: {{Audio Item CID}}

{{BINARY AUDIO ATTACHMENT}}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |
| dialogRequestId | A unique ID used to correlate directives sent in response to a specific [Recognize](https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/reference/speechrecognizer" \l "recognize) event. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| url | A unique identifier for audio content. The URL always follows the prefix cid:.  ****Example:**** cid:{{STRING}} | string |
| format | Provides the format of returned audio.  ****Accepted value:**** "AUDIO\_MPEG" | string |
| token | An opaque token that represents the current text-to-speech (TTS) object. | string |

### **SpeechStarted Event**

The SpeechStarted event should be sent to TVS after your client processes the [Speak](https://developer.amazon.com/zh/docs/alexa-voice-service/speechsynthesizer.html" \l "speak) directive and begins playback of synthesized speech.

****Sample Message****

{

"event": {

"header": {

"namespace": "SpeechSynthesizer",

"name": "SpeechStarted",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | The opaque token provided by the Speak directive. | string |

### **SpeechFinished Event**

The SpeechFinished event must be sent after your client processes a Speak directive and TVS TTS is fully rendered to the user. If playback is not finished, for example a user interrupts TVS TTS with "DingDangDingDang, stop", then SpeechFinished is not sent.

****Sample Message****

{

"event": {

"header": {

"namespace": "SpeechSynthesizer",

"name": "SpeechFinished",

"messageId": "{{STRING}}"

},

"payload": {

"token": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| token | The opaque token provided by the Speak directive. | string |

## System interface

The System interface exposes events which span multiple client components.

### **SynchronizeState Event**

The SynchronizeState event must be sent to update TVS on the state of all product components when a new connection is established.

****Sample Message****

{

"context": [

// This is an array of context objects that are used to communicate the

// state of all client components to TVS. See Context for details.

],

"event": {

"header": {

"namespace": "System",

"name": "SynchronizeState",

"messageId": "{{STRING}}"

},

"payload": {

}

}

}

****Context****

This event requires your product to report the status of all client component states to TVS in the context object.

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

An empty payload should be sent.

### **UserInactivityReport Event**

This event must be sent after an hour of inactivity, and every hour after that until a user action is taken. This provides TVS with the duration since the last user activity was detected. A user activity is defined as an action that confirms a user is in the presence of the product, such as interacting with on-product buttons, speaking with TVS, or using a GUI affordance. After a user activity is detected, the timer used to track inactivity must be reset to 0.

**Tip:** The value provided for inactiveTimeInSeconds should always be a multiple of 3600 (1 hour). For example, after 4 hours of inactivity the value would be 14400.

****Sample Message****

{

"event": {

"header": {

"namespace": "System",

"name": "UserInactivityReport",

"messageId": "{{STRING}}"

},

"payload": {

"inactiveTimeInSeconds": {{LONG}}

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| inactiveTimeInSeconds | Time in seconds since the last user interaction. | long |

### **ResetUserInactivity Directive**

The ResetUserInactivity directive is sent to your client to reset the inactivity timer used by [UserInactivityReport](https://developer.amazon.com/zh/docs/alexa-voice-service/system.html" \l "userinactivityreport). For example, a user interaction on the Tencent DingDang app would trigger this directive.

****Sample Message****

{

"directive": {

"header": {

"namespace": "System",

"name": "ResetUserInactivity",

"messageId": "{{STRING}}"

},

"payload": {

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

### **SetEndpoint Directive**

The SetEndpoint directive instructs a client to change endpoints when the following conditions are met:

* A user's ****country/region settings**** are not supported by the endpoint they have connected to. For example, if a user's ****current country/region**** is set to the United Kingdom (UK) in [Manage Your Content and Devices](https://www.amazon.com/gp/help/customer/display.html?nodeId=202205900)and the client connects to the United States (US) endpoint, a SetEndpoint directive will be sent instructing the client to connect to the endpoint that supports the UK.
* A user changes their ****country/region settings**** (or address). For example, if a user connected to the US endpoint changes their ****current country/region**** from the US to the UK, a SetEndpoint directive will be sent instructing the client to connect to the endpoint that supports the UK.

**Important:** Failure to switch endpoints may result in a user not having access to custom preferences and country or region specific content.

****Sample Message****

{

"directive": {

"header": {

"namespace": "System",

"name": "SetEndpoint",

"messageId": "{{STRING}}"

},

"payload": {

"endpoint": "{{STRING}}"

}

}

}

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| endpoint | The TVS endpoint URL that supports your user's country/region settings. The endpoint URL may include the protocol and/or port. ****For example****: https://tvs.html5.qq.com | string |

### **SoftwareInfo Event**

This event communicates your product's software information to TVS, such as firmware version. It must be sent in these scenarios:

* For products with persistent memory, the event must be sent on the product's initial boot and whenever the firmware version is updated.
* For products ****without**** persistent memory, the event must be sent on each boot/reboot.
* When a ReportSoftwareInfo directive is received.

If the event is successfully processed, the product will receive a 204 HTTP status code with an empty body. If the event is not processed the product will receive a 500 HTTP status code and an [Exception Message](https://developer.amazon.com/docs/alexa-voice-service/exceptions.html) from TVS.

****Sample Message****

{

"event": {

"header": {

"namespace": "System",

"name": "SoftwareInfo",

"messageId": "{{STRING}}"

},

"payload": {

"firmwareVersion": "{{STRING}}"

}

}

}

****Header Parameter****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A universally unique identifier (UUID) generated to the [RFC 4122](https://tools.ietf.org/html/rfc4122) specification. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| firmwareVersion | A positive signed 32-bit integer represented as a string. If an invalid value is sent to TVS, an HTTP 400 status code is returned to your client. ****IMPORTANT****: "0" is not a valid firmware version.   |  |  | | --- | --- | | Valid | Invalid | | * "123" * "8701" * "20170207" | * "50.3" * "tvs-123.4x" * "tsk.201-(1.23.4-test)" | | string |

### **ReportSoftwareInfo Directive**

This directive instructs your product to report current software information to TVS using the SoftwareInfo event.

****Sample Message****

{

"directive": {

"header": {

"namespace": "System",

"name": "ReportSoftwareInfo",

"messageId": "{{STRING}}"

},

"payload": {

}

}

}

****Header Parameter****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A universally unique identifier (UUID) generated to the [RFC 4122](https://tools.ietf.org/html/rfc4122) specification. | string |

### **ExceptionEncountered Event**

Your client must send this event when it is unable to execute a directive from TVS.

****Sample Message****

{

"context": [

// This is an array of context objects that are used to communicate the

// state of all client components to TVS. See Context for details.

],

"event": {

"header": {

"namespace": "System",

"name": "ExceptionEncountered",

"messageId": "{{STRING}}"

},

"payload": {

"unparsedDirective": "{{STRING}}",

"error": {

"type": "{{STRING}}"

"message": "{{STRING}}"

}

}

}

}

****Context****

This event requires your product to report the status of all client component states to TVS in the context object.

****Header Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| messageId | A unique ID used to represent a specific message. | string |

****Payload Parameters****

|  |  |  |
| --- | --- | --- |
| ****Parameter**** | ****Description**** | ****Type**** |
| unparsedDirective | When unable to execute a directive, your client must return the directive to TVS as a string. | string |
| error | Key/value pairs for error. | object |
| error.type | An error your client must return to TVS when unable to execute a directive. | string |
| error.message | Additional error details for logging and troubleshooting. | string |

****Error Types****

|  |  |
| --- | --- |
| ****Error Type**** | ****Description**** |
| UNEXPECTED\_INFORMATION\_RECEIVED | The directive sent to your client was malformed or the payload does not conform to the directive specification. |
| INTERNAL\_ERROR | An error occurred while the device was handling the directive and the error does not fall into the specified categories. |