Orchestration Server Developer's Guide

External Interfaces

12/14/2025

External Interfaces

Contents

The orchestration platform has a set of RESTFul Web 2.0 Web Services APIs. These APIs allow the following interaction with SCXML sessions:

- **Start SCXML Session** - This action starts a new orchestration application instance (session).
- **Stop SCXML Session** - This action terminates a given orchestration application instance (session).
- **Publish Event to SCXML Session** - This action allows an external system to send an event to a given orchestration session.
- **Send a Request to SCXML Session** - This action allows an external system to send a request to a given session to be processd. The session will respond to this request with the `<response>` element.
- **Query SCXML Session** - This action gets the requested orchestration session data. This can be used to get current session data on any session.

> **Tip**
>
> (Starting with 8.1.400.55), the `/ors/help` method provides help for ORS REST APIs.

These external interfaces may have equivalent functionality in SCXML or Orchestration extensions. The following table reflects the mapping between the two.

External RESTful APIs	SCXML or Functional Module Function
`http://<server:port>/scxml/session/start`	`<session:start>`
`http://<server:port>/scxml/session/<session id>/terminate`	`<session:terminate>`
`http://<server:port>/scxml/session/<session id>/event/<name>[?<parameters>]`	`<scxml:send>`
`http://<server:port>/scxml/session/<session id>/request/<name>[?<parameters>]`	`<session:fetch>`
`http://<server:port>/scxml/session/<session id>/query`	none

Start SCXML Session

This API starts a new SCXML session for a specific application. The "src" parameter is mandatory. All of the other parameters are optional.

http://<server:port>/scxml/session/start		
HTTP Verbs	PUT	Not used
	POST	Used to start a given session. (supported Content-Type - application/x-www-form-urlencoded). Starting with ORS 8.1.400.30, application/json is supported as a new value for Content-Type.
	DELETE	Not used
	GET	Not used
URI-Variable Elements	none	
Request-URI Parameters	none	
Document Body - using application/x-www-form-urlencoded	src	URL of the SCXML document to use for the new session.
	idealtime	A dateTime value which will represent the date and time that this session is to be started. This value should be the time as returned by the ECMAScript Date(...).getTime() function, which is given in the number of milliseconds since 00:00:00 UTC on January 1, 1970.
	request-specific	These are request-specific parameters. These parameters will be put in the appropriate session data items (<data>) when the session is initiated if the name of the parameter matches the ID attribute of the <data> element. For example, if you have the following parameters, p1=12355, p2=abcd, then <data id="p1> and <data id="p2"> will be set to the corresponding values. If a parameter value does not match the ID of a data item, it will be thrown away. In addition, there will be a parameter which specifies the content type for the parameters and it will be set to "application/x-www-form-urlencoded".
	results	A body parameter value which will represent a callback URL that accepts the results of a scheduled session start, whether it has positive or negative results. This URL will be invoked with the HTTP POST method. The content of the document body will be the following: • Type - This is the type of

http://<server:port>/scxml/session/start		
		response - positive or negative. • Reason - This is the reason why the response was generated. • sessionid - This is the ID of the session that is being started. • server - This is the URL of the server that can be used to invoke other requests on the same session. (Question: Should this be merged with the sessionid?)
Positive Response (200 Response)	ID	The identifier of the newly created session.
Negative Response	HTTP Error Code	HTTP 4xx
Example	POST http://<server:port>/scxml/session/start . . . Content-type: application/x-www-form-urlencoded . . . src=http://appserver/appname.scxml	

Stop SCXML Session

This API terminates a SCXML session.

http://<server:port>/scxml/session/<session id>/terminate		
HTTP Verbs	PUT	Not used
	POST	Stops an existing session.
	DELETE	Not used
	GET	Not used
URI-Variable Elements	session id	This identifies the session which is to be stopped.

http://<server:port>/scxml/session/<session id>/terminate		
Document Body	none	none
Positive Response (200 Response)	OK	OK
Negative Response	HTTP error code	HTTP 4xx
Example	POST http://<server:port>/scxml/session/1234567/terminate	

Query SCXML Session

This API queries a given SCXML session's data.

http://<server:port>/scxml/session/<session ids>/query		
HTTP Verbs	PUT	Not used
	POST	Not used
	DELETE	Not used
	GET	Query a set of existing sessions.
URI-Variable Elements	session ids	This identifies the set of sessions which are to be queried. The session ids are separated by a "," For example, <server:port>/scxml/session/123456,345677,66778898 . Currently, we only support a list of one session id.
Request-URI Parameters	none	
Document Body	none	none
Positive Response (200 Response)	sessionData	This is a list of sessions and their associated data. The following is the JSON-formatted set of session data which will be returned for each session in the list: • Session ID • URL of the application • Name - _name attribute

http://<server:port>/scxml/session/<session ids>/query		
		• Type - _type attribute • Current states • Current events • _data properties (application-related data) • _genesys properties (functional module-related data)
Negative Response	HTTP error code	404 Not found
Example	GET http://<server:port>/scxml/session/1234567/query	

Send Request to SCXML Session

This API sends a request to the SCXML session to process.

http://<server:port>/scxml/session/<session id>/request/<name>		
HTTP Verbs	PUT	Not used
	POST	Send a request to an existing session.
	DELETE	Not used
	GET	Not used
URI-Variable Elements	session id	This is the session which the request is targeted for.
	name	This is the name of the request which is to be performed by the SCXML session when it receives the corresponding request event. This value will be the name of the SCXML event which the SCXML session will process.
Request-URI Parameters	none	
Document Body - can be in any of the following encodings - application/x-www-form-urlencoded - application/json - text/xml	request-specific	These are request-specific parameters. These parameters will be put into the SCXML event at the following location: _event.data.param.xxx. For example, if

http://<server:port>/scxml/session/<session id>/request/<name>		
		you have the following body parameters, p1=12355, p2=abcd, then the following will be the structure in the event: _event.data.param.p1 and _event.data.param.p2. In addition, there will be a parameter which specifies the content type for the parameters and it will be set based HTTP Content-Type element value. For details see the [[|API Parameter passing]] section.
	request identifier	This API-related parameter is generated by the orchestration platform when it gets the HTTP request. It is used to correlate the requests and the corresponding responses for a given session. This identifier is put in the sendid property of the request's SCXML event (that is, _event.sendid) when sent to the application. This identifier must be used in the corresponding `<response>` element.
Positive Response (200 Response)	results	This is a set of data, based on the results of the request. It is request-specific in its content. The SCXML session sends this data via the [[|<response>]] element.
	headers	This is a collection of key-value pairs, representing a select group of request headers obtained from the HTTP request. It is request-specific in its content. This data can be accessed in the SCXML event at the following location: _event.data.headers The list of retrievable headers are as follows: • HTTP_METHOD, HTTP_VERSION, HTTP_REQUEST_URI, ACCEPT, DATE, USER-AGENT, CONNECTION, ACCEPT-LANGUAGE, REFERER, IF-MODIFIED-SINCE, FROM, MIME-VERSION, PRAGMA, AUTHORIZATION, CONTENT-LENGTH, CONTENT-TYPE, CONTENT-ENCODING
Negative Response	HTTP error code	HTTP 4xx
Example	`POST http://<server:port>/scxml/session/1234567/request/getxData`	

http://<server:port>/scxml/session/<session id>/request/<name>	
	. . . Content-type: application/x-www-form-urlencoded . . . parm1=john The SCXML session must have the following SCXML snippet somewhere in its definition. This type of request processing SCXML snippet is probably best placed as a global document event handler. <transition event="getxData" cond="_event.data.param.parm1 = ''" > <ws:response requestid="_event.sendid" type="negative" resultcode="invalidparameter"/> </transition> <transition event="getxData" cond="_event.data.param.parm1 != ''" > <script> var rdata = _getdata(_event.data.param.parm1); </script> <ws:response requestid="_event.sendid" > <param name="results" expr="rdata"/> </ws:response> </transition>
Special Considerations	It is recommended that the orchestration logic for processing this event should be processed within the executable content of the <transition> element that receives the request event. This includes using the <response> element to respond to the request. If the processing of the request requires more complicated processing (for example, it must transition to a sub-state model for processing), then the application must copy all the necessary request data (for example, _event.sendid) from the event and put it into the appropriate global variables so that the sub-state model can use it to process the request. This is because the event data becomes invalid after the transition element has been processed.

Publish Event to SCXML Session

This API sends an event to an existing session.

http://<server:port>/scxml/session/<session id>/event/<name>		
HTTP Verbs	PUT	Not used
	POST	Send an event to an existing session.
	DELETE	Not used
	GET	Not used

http://<server:port>/scxml/session/<session id>/event/<name>		
URI-Variable Elements	session id	This is the session which the event is targeted for.
	name	This is the name of the event which is to be sent to the SCXML session. This value will be the name of the SCXML event which the SCXML session will process.
Request-URI Parameters	none	
Document Body - can be in any of the following encodings - application/x-www-form-urlencoded - application/json - text/xml	request-specific	These are event-specific parameters. These parameters will be put into the SCXML event at the following location: _event.data.param.xxx. For example, if you have the following body parameters, p1=12355, p2=abcd, then the following will be the structure in the event: _event.data.param.p1 and _event.data.param.p2. In addition, there will be a parameter which specifies the content type for the parameters and it will be set based HTTP Content-Type element value. For details see API Parameter Passing.
Positive Response (200 Response)	none	**Note:** (Since 8.1.200.50) Behaviour depends on configuration option orchestration/webfm-event-hold-response (default = true). When [Orchestration/ webfm_event_hold_response] is true, no response is given until the event is processed. The response status code depends on how the event is processed: • Event is processed and transition is taken - OK 200 Response • Event is processed and no transition is taken - No Content 204 Response When [Orchestration/ webfm_event_hold_response] is false, 200 Response is provided immediately. Prior to 8.1.200.50, 200 Response is provided immediately.
	headers	This is a collection of key-value pairs, representing a select group of request headers obtained from the HTTP request. It is request-specific in its content. This data can be accessed in the SCXML event at the following location: _event.data.headers The list of retrievable headers are as

http://<server:port>/scxml/session/<session id>/event/<name>		
		follows: • HTTP_METHOD, HTTP_VERSION, HTTP_REQUEST_URI, ACCEPT, DATE, USER-AGENT, CONNECTION, ACCEPT-LANGUAGE, REFERER, IF-MODIFIED-SINCE, FROM, MIME-VERSION, PRAGMA, AUTHORIZATION, CONTENT-LENGTH, CONTENT-TYPE, CONTENT-ENCODING
Negative Response	HTTP error code	HTTP 4xx
Example	POST http://<server:port>/scxml/session/1234567/event/xisDone . . . Content-type: application/x-www-form-urlencoded . . . parm1=john The SCXML session must have the following SCXML snippet somewhere in its definition. This type of request-processing SCXML snippet is probably best placed as a global document event handler. <transition event="xisDone" target="continuewithY" > <! - Do some processing here ..../> </transition>	

Orchestration Platform Status

The status of the orchestration platform can be obtained simply by using a GET HTTP request. The following are the URLs for different types of platform status. The results from the request are in XML.

URL	Type	Results
<Orch platform server>:<port>	Basic data	• Version • Start time • Running time
<Orch platform server>:<port>/server?cfgStatistics	Configuration data	

URL	Type	Results
<Orch platform server>:<port>/server?activeCalls	Interaction data	
<Orch platform server>:<port>/server?scxmlStat	SCXML engine data	
<Orch platform server>:<port>/serverx?activeApplications	SCXML Application data	For each active application, the following information is provided: • URL • Name • startedSessions • endedSessions • abortedSessions

API Parameter Passing

Parameters are passed from the Web 2.0 APIs to the SCXML session using two basic mechanisms. These mechanisms are **mutually exclusive**:

URL-Encoded Parameters

URL-encoded parameters can be specified in a request. Parameters may be specified either in the URL for GET based APIs:

```
GET http://<server:port>/scxml/session/<session id>/event/<name>?param1=1¶m2=2
```

Or in the document body for POST and PUT based APIs:

```
POST http://<server:port>/scxml/session/<session id>/event/<name>
.. . .
Content-type: application/x-www-form-urlencoded
. . .
param1=1¶m2=2
```

In both cases, URL encoded parameters will be translated to properties of the SCXML event:

```
_event.data.param.param1 = 1
_event.data.param.param2 = 2
```

The _event.data.paramtype will be set to application/x-www-form-urlencoded. In the case of the `[http:// http://]<server:port>/scxml/session/start` API (Start Session), the following processing will be done:

- • If the name of the parameter matches the 'id' of a <data> element in the data model of the started session, then the value of the parameter will replace the value of the corresponding <data>

element.

- If the names do not match, the value of the parameter will not be added to the started session's data model.

Document Body-Encoded Parameters

These parameters are specified in the document body using the following supported content types:

- **application/json** - In this case, the body of the request contains JSON-encoded data. The whole body of the request is passed to the SXCML session as a value of the "_event.data.param" event property. In addition, the "_event.data.paramtype" property will be set to "application/json", based on the HTTP Content-Type element. For example:

```
POST http://<server:port>/scxml/session/<session id>/event/<name>
...
Content-Type=application/json
Content-Length=xx
...
param="{
    "firstName": "John",
    "lastName": "Smith",
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": 10021
    },
    "phoneNumbers": [
        "212 555-1234",
        "646 555-4567"
    ]}"
```

An SCXML application may process these parameters by evaluating JSON text to ECMAScript objects. In the case of the [http:// http://]<server:port>/scxml/session/start API (Start Session), the following processing will be done:

- If the name of the parameter matches the 'id' of a <data> element in the data model of the started session, then the value of the parameter will replace the value of the corresponding <data> element.
- If the names do not match, the value of the parameter will not be added to the started session's data model.

text/xml - In this case, the body of the request contains XML-encoded data. The whole body of the request is passed to the SXCML session as a value of the "_event.data.param" event property. In addition, the "_event.data.paramtype" property will be set to "text/xml". For example:

```
POST http://<server:port>/scxml/session/<session id>/request/<name>
...
Content-Type=text/xml
Content-Length=xx
...
param="
<findCar>
    <make>Dodge</make>
    <model>Daytona</model>
</findCar>
"
```

An SCXML application may process this parameter using ECMAScript XML capabilities. In the case of the http://server:port/scxml/session/start API (Start Session), the following processing will be done:

- If the name of the parameter matches the 'id' of a <data> element in the data model of the started session, then the value of the parameter will replace the value of the corresponding <data> element.
- If the names do not match, the value of the parameter will not be added to the started session's data model.