API Reference

Genesys Web Engagement 8.1.1

1/3/2022

Table of Contents

Genesys Web Engagement API Reference

Purpose: To list the Reference Guides available for Genesys Web Engagement

Monitoring JavaScript Reference

You can use this JavaScript Reference to define Business Events sent by your web pages to the Web Engagement Frontend Server. These events are independent from the set of conditions and definitions used to create events with the Monitoring DSL Reference. See Monitoring JavaScript Reference.

Monitoring DSL Reference

You can use the Monitoring DSL Reference to define conditions which enable the Web Engagement Monitoring Agents to submit BUSINESS events to the Web Engagement Frontend Server.

- To learn more about the Monitoring Agent, see Architecture.
- To learn more about the Advanced Model, which include DSL event conditions and definitions, see Advanced Engagement.

History REST Reference

You can use the History REST Reference to create REST queries and access data stored in the Genesys Web Engagement Backend Server. The Backend Database stores all the histories related to the customers visits: events, visited pages, sessions, user information. See History REST for further details.

Monitoring JavaScript

Introduction

The Monitoring JavaScript API is available through the JavaScript libraries provided by the Web Engagement Frontend Server at runtime (see Architecture for details.) By using this API in your web pages, you can submit events to the Genesys Web Engagement Frontend Server, independently from the set of events and conditions defined in the DSL files loaded by the browser's Monitoring Agents. The design model for the Monitoring JavaScript API is highly flexible, and its use can be extended well beyond the common model of user-triggered events—the design decision is up to you. You can submit `UserInfo`, `SignIn`, `SignOut`, and your own custom business events using this API.

> **Important**
> All methods and parameters are **case sensitive**.

API Entry Point

The entry point for this API is the global `_gt` (Genesys Tracker) object which implements a `push()` method to push events to the Frontend Server.

The `gt.push()` method takes into parameter a JSON array, which can contain any type of information and follows this syntax:

```
_gt.push(['<commandName>', { <parameter-1>: 'value-1', <parameter-2> : '<value-2>', ...,
<parameter-n>: '<value-n>'}]);
```

where:

- `<commandName>` is the name of the event command;
- `<parameter-i>` is the name of a command parameter;
- `<value-i>` is the value of a command parameter.

For instance, the following example shows how to use the `event` command to create a BUSINESS event and record a customer interaction with an `Add To Cart` link.

```
<a href="#" onClick="_gt.push(['event', { eventName: 'AddToCart',
            productName : 'Sony', productModel: 'JVB72',
            productPrice: '1000$'}]);">Add to Cart</a>
```

In this scenario, if the customer clicks on `Add to Cart` link, the `_gt.push()` method sends the `AddToCart` BUSINESS event to the Web Engagement Frontend

Server. The event includes information about the product name, model, and price.

Possible Event Commands

sendUserInfo

Description: Sends a SYSTEM event including customer information. You should send this event only if the website has authenticated the user.

Parameters for the **sendUserInfo** command

Name	Value	Mandatory	Description
userID	string	yes	The identification ID or the user. For instance, the user account name or the e-mail address.
<customParameter>	<customValue>	no	Any custom parameter containing user information.

Example with mandatory parameters only:

```
_gt.push(['sendUserInfo', {userID: 'user@genesyslab.com'}]);
```

Example with additional user information

```
_gt.push(['sendUserInfo', {userID: 'user@genesyslab.com',
name: 'Bob',
sex: 'male',
age: 30}]);
```

sendSignIn

Description: Creates and sends the SignIn event. This SYSTEM event should be sent when the user authenticates.

Parameters for the **sendSignIn** command

Name	Value	Mandatory	Description
userID	string	yes	The identification ID or the user. For instance, the user account name or the e-mail address.
<customParameter>	<customValue>	no	Any custom parameter containing the user information.

Example with mandatory parameters only:

```
_gt.push(['sendSignIn', {userID: 'user@genesyslab.com'}]);
```

Example with additional parameters:

```
_gt.push(['sendSignIn', {userID: 'user@genesyslab.com',
name: 'Bob',
sex: 'male',
age: 30}]);
```

sendSignOut

Description: Creates and sends the `SignOut` event for the current user. This event should be sent if the user performs a logout on the website or as soon as possible after the logout action was done.

Parameters for the **sendSignOut** command

Name	Value	Mandatory	Description
userID	string	no	The identification ID or the user. For instance, the user account name or the e-mail address. **Note:** Genesys recommends the use of this parameter even if it is not mandatory.
<customParameter>	<customValue>	no	Any custom parameter containing user information.

Example with no parameter:

```
_gt.push(['sendSignOut']);
```

Example with additional parameters:

```
_gt.push(['sendSignOut', { userID: 'user@genesyslab.com'}]);
```

event

Description: Sends BUSINESS events to the server.

Parameters for the **event** command

Name	Value	Mandatory	Description
eventName	string	yes	The identification name of the event. This field is equivalent to the name parameter of the DSL <event> element.
<customParameter>	<customValue>	no	Any custom parameter.

Example with mandatory parameters only:

```
_gt.push(['event', { eventName: 'SomeEvent'}]);
```

Example with additional parameters:

```
_gt.push(['event', { eventName: 'AddToCart',
productName : 'Sony',
productModel: 'JVB72',
productPrice: '1000$'}]);
```

Monitoring DSL

Description

The Web Engagement Monitoring Agent DSL Reference lists all the DSL elements that you can use to define Business Events in the Monitoring Agent Component. For details about implementation, read Creating Business Events. The DSL specifies the document elements to monitor, the events to send to the Frontend Server, and the data to include with those events.

```
<?xmlversion="1.0"encoding="utf-8"?>
<properties debug="false">
    <events>
        <event id="AddToCartEvent" name="AddToCart">
            <trigger name="AddToCartTrigger" element="img.bdt-addToCart" action="click"
url="http://www.MySite.com/" count="1">
                <val name="productName"
value="$(event.target).parents('div.hproduct').find('h3.name a').text()"/>
                <val name="productModel"
value="$(event.target).parents('div.hproduct').find('span.model')"/>
                <val name="productSKU"
value="$(event.target).parents('div.hproduct').find('span.sku').text()"/>
                <val name="productPrice"
value="$(event.target).parents('div.hproduct').find('h4.price').text()"/>
            </trigger>
    </event>

    <event id="SearchEvent" name="Search">
        <trigger name="GoSearchClick" element="td#gobutton input" action="click"
url="http://www.MySite.com" count="1"/>
        <val name="searchString" value="$('input.searchfield:text').val();"/>
    </event>
    </events>
</properties>
```

<properties> (mandatory)

The `<properties>` element is the main root element of the DSL file. It has an optional debug attribute and a mandatory `<events>` child.

debug (optional)

The debug attribute enables debugging in the browser by setting its value to the JavaScript Boolean `true`. The debugging information opens a pop-up window and shows the JSON serialized event data for the business events before they are sent to the Frontend Server.

Note: In some browsers, using the debug attribute can affect the performance of the Frontend Server by delaying the event dispatch.

<events>

The <events> element contains a list of all the business events that can be generated during monitoring. These business events are captured in the <event> child element.

<event>

The <event> element contains mandatory `id` and `name` attributes, and an optional `condition` attribute. An <event> must also have one or more <trigger> children, which define the conditions that must be matched to generate an event.

Note: If the <trigger> child is omitted, the event will never be generated.

id (mandatory)

The `id` is the internal identifier for the event. This is must be unique across the DSL file, to make a distinction between the events sent by the browser.

name (mandatory)

The `name` is sent to the Frontend Server. A DSL file may contain several <event> elements with identical values for `name`, but with different values for `id`. For example, if your website includes a search form, you can submit this form by clicking on the 'search' button or by pressing the 'enter' key. Inside the browser, the click and key press events are clearly distinct, but are not relevant for the Frontend Server.

The following example shows how to create two business events which return the same event name to the Frontend Server:

```
<?xmlversion="1.0"encoding="utf-8"?>
<properties debug="false">
    <events>
        <event id="SearchEvent" name="Search">
            …
        </event>
        <event id="SearchKeyDownEvent" name="Search">
            …
        </event>
    </events>
</properties>
```

condition (optional)

The `condition` attribute is a JavaScript Boolean expression. If it is present, the event's triggers will be installed in the page if the `condition` evaluates to `true`.

The following example creates a business event with a timer which can be triggered only if the text inside the <h1> tag on the page is "Compare":

```
<event id="InactivityTimeout4CompareProductsEvent" name="InactivityTimeout4CompareProducts"
condition="$('h1').text() == 'Compare'">
```

```
    <trigger name="InactivityTimeout4CompareProductsTrigger" element="" action="timer:10000"
type="timeout"
       url="http://www.MySite.com/site/olspage.jsp" count="1"/>
      …
</event>
```

Since the event (in this case 'InactivityTimeout4CompareProductsEvent') will never be generated if its triggers are not installed, the `condition` attribute allows you to place conditions on any feature of the environment that can be tested by a JavaScript Boolean expression, in order to monitor and generate events.

<trigger> (mandatory child element)

The `<trigger>` element defines the conditions that must be matched to generate business events, as well as the data to be included with the event. If several triggers are part of the event definition, they must all match to raise the business event. If each trigger matches a different DOM event in the browser, then the set of triggers specifies a series of web events that must occur before the parent business event is submitted to the Frontend Server.

The `<trigger>` element has mandatory `name`, `element`, and `action` attributes, and optional `url` and `count` attributes. It can have and 0 or more `<val>` children.

name (mandatory)

This attribute specifies the name of the trigger. It must be unique in the parent `<event>` element. If an `<event>` element has multiple triggers, they must all have different names.

element (mandatory)

The `element` attribute specifies the document's DOM element to which the trigger should be attached. The value of `element` should be a jQuery selector. For details on jQuery selectors, see http://api.jquery.com/category/selectors/. If the `action` attribute is set to `timer:nnn`, then `element` can be set to null.

action (mandatory)

The `action` specifies the DOM event to track. The trigger is matched if this DOM event is targeted to the DOM element specified by the `element` attribute.

You can specify standard DOM events for the `action`:

- Browser Events
- Document Loading
- Keyboard Events
- Mouse Events
- Form Events

In addition to the standard DOM events, the DSL supports the following two values: `timer` and `enterpress`.

The enterpress event signals that the user has pressed the 'enter' key. It is more specific than the standard DOM 'keypress' event, which is raised when any key is pressed.

The following example adds a value named "searchString" when the "Search" event is generated and sent to the Frontend Server. The value is the string entered in the "searchfield" textbox.

```
<event id="SearchKeyDownEvent" name="Search">
    <trigger name="SearchKeyDown" element="input.searchfield:text" action="enterpress"
url="http://www.MySite.com" count="1"/>
    <val name="searchString" value="$('input.searchfield:text').val();"/>
</event>
```

Setting the action attribute to timer allows you to specify the time interval, in milliseconds. For example, action="timer:10000" specifies a 10-second timer. If you set the value of action="timer:nnn", you must also include the type attribute to specify how the timer works.

type (mandatory when action="timer:nnn")

The type attribute is mandatory when action="timer:nnn". The type can have a value of either timeout or nomove, which specifies how the timer action works.

If type="timeout", the timer interval begins after the page is loaded. If type="nomove", the timer resets each time the user moves the mouse.

In the following example, the "InactivityTimeout" event is generated after the user has been inactive for 10 seconds.

```
<event id="InactivityTimeout4CompareProductsEvent" name="InactivityTimeout"
condition="$('h1').text() == 'Compare'">
    <trigger name="InactivityTimeout" element="" action="timer:10000" type="nomove"
       url="http://www.MySite.com/site/olspage.jsp" count="1"/>
    <val name="products" value="…" />
</event>
```

If type="timeout" was specified instead, the event would be generated 10 seconds after the page was loaded.

url (optional)

The url attribute defines the URL of a specific page which raises the business event. The business event is not submitted if the current document's URL does not macth the URL parameter.

count (optional)

The count attribute specifies how many times the trigger needs to be matched before the event is generated and sent to the Frontend Server.

<val>

The <val> element can be used to add data to the business event. You can have 0 or more <val> elements; each instance adds a field to the business event. If <val> is a child of <trigger>, it can also have access to the DOM event matched by the trigger.

name (mandatory)

The name attribute is the name of the value in the generated business event. The name of each val must be unique inside a parent event. The name will be added to the generated business event's data, along with the corresponding value attribute.

The following example adds a value named "searchString" when the "Search" event is generated and sent to the Frontend Server.

```
<event id="SearchEvent" name="Search">
    <trigger name="GoSearchClick" element="td#gobutton input" action="click" url=" "
count="1"/>
    <val name="searchString" value="$('input.searchfield:text').val();"/>
</event>
```

The following output is an example of an event submitted to the Frontend Server when the visitor enters "my search string" in the search box and clicks the search button. The "eventName" parameter is the name attribute of the <event> element. The "searchString" parameter is the name attribute of the <val> element and "my search string" is its value. The additional fields are generated automatically by the DSL code:

```
{
   "data":{
      "searchString":"my search string"
   },
   "eventType":"BUSINESS",
   "eventName":"Search",
   "eventID":"D88B2FF5A9C24095837CF105FB6D5CF9",
   "pageID":"A9D1E9265D444351876C13D6C5FA5FAD",
   "timestamp":1309962580226,
   "globalVisitID":"7E67BA9701124F738CAC80DDFEA1D705",
   "visitID":"4608DD210B034AC18C65C2C2275CD8B6",
   "userID":"",
   "url":"http://www.MySite.com/site/",
   "category":""
}
```

value (optional)

The value attribute specifies the value to associate with the name attribute in the field of the generated event. Its value can be any JavaScript code which returns a serializable object.

The following example tracks search events and includes the search string in the event when it is sent to the Frontend Server. In this example, there is only one search input box on the page:

```
<event id="SearchEvent" name="Search">
    <trigger name="GoSearchClick" element="td#gobutton input" action="click"
url="http://www.MySite.com" count="1" />
    <val name="searchString" value="$('input.searchfield:text').val();"/>
</event>
```

In the following example, the "AddToCart" event is tracked, including information about the product that was added: name, model, SKU, and price. Tracking by clicking on the "add to cart" button does not provide information about which button was clicked and which product was added to the cart. To get this information, you need to use the DOM event object: "event.target" identifies the clicked button, which can provide information related to the product.

```
<?xml version="1.0"encoding="UTF-8"?>
```

```
<event id="AddToCartEvent" name="AddToCart">
    <trigger name="AddToCartTrigger" element="div.info-side img.bdt-addToCart" action="click"
url="http://www.MySite.com"  count="1">
        <val name="productName" value="$(event.target).parents('div.hproduct').find('h3.name
a').text()"/>
        <val name="productModel"
value="$(event.target).parents('div.hproduct').find('span.model').text()"/>
        <val name="productSKU"
value="$(event.target).parents('div.hproduct').find('span.sku').text()"/>
        <val name="productPrice"
value="$(event.target).parents('div.hproduct').find('h4.price').text().replace('Sale:', )"/>
    </trigger>
</event>
```

History REST

Description

The Web Engagement History REST Reference lists all the RESTful elements that you can use to access the Web History stored by the Genesys Web Engagement Backend Server.

Resource Access Template

To access resources, you must respect the following syntax:

```
Method <schema>://<server>:<port>/backend/data[/<resourceName>][/<resourceId>][/<sub-resourceName>]
```

where:

- `<method>` is `GET` or `POST` ;
- `<schema>` is `http` or `https`;
- `<server>` is the hostname or the IP address of the Web Engagement Backend Server;
- `<port>` is the port of the Web Engagement Backend Server;
- `<resourceName>` is the name of the requested resource;
- `<resourceId>` is the identifier of the requested resource;
- `<sub-resourceName>` is the name of a sub-resource.

For instance, to retrieve the list of identities on MyHostname, you can use the following request:

```
GET http://MyHostname:9081/backend/data/identities
```

For instance, to retrieve a given identity, you need its identifier:

```
GET http://MyHostname:9081/backend/data/identities/pat.thompsom@genesyslab.com
```

For retrieving the pages of the user pat.thompsom@genesyslab.com, you can use the following request:

```
GET http://MyHostname:9081/backend/data/identities/pat.thompsom@genesyslab.com/pages
```

Contents

The list of available resources is the following:

- Event Resource
- Identity Resource
- Page Resource
- Session Resource
- Visit Resource

Note that each resource chapter lists all the operations available with the resource's path, not all the operations related to the resource. For instance, the Event Resource chapter lists all the methods associated with the **backend/data/events/** path. Additional methods to retrieve events for a given identity are available in the Identity Resource chapter. You can also use the Operations Index to get the complete list of available operations.

HTTP Response Codes and Errors

Introduction

The Genesys Web Engagement Backend Server returns HTTP status codes and messages for every operation, in the requested format. Status codes match standard HTTP codes, but messages can differ and provide additional details included in the header of the response. Additional result and error codes may be returned due to external web servers and layers involved in your operations.

Successful Result

A successful response to a request is marked by HTTP Status Code 200 (OK). In that case, your application may get additional information in the header and the body of the response. Refer to the Response section of your operation's page to get the detailed list of returned information. Questions about the returned content can be submitted as comments in this wiki. The following table lists the standard HTTP codes used by Genesys Web Engagement Backend Server for a successful response.

Successful results

Code	Title	Description
200	OK	Success!
204	No Content	For "filtered collection read" requests only. The request was correct and successful but the server have no appropriate entities to return.

Errors

For responses with HTTP status code 4xx or 5xx, the response body contains an application-specific description of the error instead of a representation of the requested resource. The following table lists the specific errors that operations can encounter. This list is not restrictive; additional error codes could be returned due to external web servers and layers involved:

Errors

Code	Title	Description
400	Bad Request	General error which can be one of the following reasons: • Missing required parameter.

Code	Title	Description
		• Parameter value of unexpected type.
401	Not Authorized	Credentials are missing or incorrect, or the given user is not allowed to execute a given service (such as an administrative service method that changes the profile schema). See Basic Access Authentication.
403<ref name="r400">Not available for now.</ref>	Forbidden	The operation is forbidden and the reason is specified in the error message.
404	Not Found	For "identified entity read" request only. The specified URI is invalid, or the requested resource (such as identity, visit, event, and so on.) does not exist.
500	Internal Server Error	An unexpected error occurred in the Web Engagement Backend Server (for instance, a runtime exception). The error message suggests to forward logs to Genesys Customer Support.
502	Bad Gateway	Returned when one or more of the backend systems required to fulfill the response (the Cassandra DB, Genesys Environment, for example) are either unavailable or returned an error.
503	Service Unavailable	Web Engagement Backend Server is unable to process the given request. Example situations include: • Too many requests.

<references />

Authentication

The Web Engagement History REST API supports the following authentication scheme:

- Basic HTTP Authentication (see http://www.ietf.org/rfc/rfc2617.txt).

HTTP authentication should be used with Secured HTTP communication (HTTPS).

Configuration

The REST API security is configured in the security section of the Web Engagement Backend Server application. The following configuration options are mandatory to enable authentication:

- auth-scheme
- user-id
- password

Note: If authentication is used, every REST API client must support that authentication type and the clients must know the authentication credentials. You must configure authentication for Interaction Workspace (see Configure Authentication in the Interaction Workspace Plug-in deployment). If your SCXML strategies use the REST interface, you must also add your authentication credentials (see Configure Authentication in the default strategy).

Basic Authentication

This authentication scheme passes unencrypted credentials, so it is unsafe unless you use a secured connection (HTTPS).

Operations Index

	Lists all the operations available.

Operations and Resources index

The REST API contains the following resources and operations:

- Event Resource (/events)
  - GET /events/${eventId}
- Identity Resource (/identities)
  - POST /identities
  - POST /identities/**${identityId}**/visits
  - GET /identities
  - GET /identities/**${identityId}**
  - GET /identities/**${identityId}**/events
  - GET /identities/**${identityId}**/pages
  - GET /identities/**${identityId}**/sessions
  - GET /identities/**${identityId}**/visits
- Page Resource (/pages)
  - GET /pages/**${pageId}**
  - GET /pages/**${pageId}**/events
  - POST /pages/**${pageId}**/events
- Session Resource (/sessions)
  - POST /sessions/**${sessionId}**/pages
  - GET /sessions/**${sessionId}**
  - GET /sessions/**${sessionId}**/pages
  - GET /sessions/**${sessionId}**/events
- Visit Resource (/visits)
  - POST /visits
  - POST /visits/**${visitId}**/events
  - POST /visits/**${visitId}**/pages

- POST /visits/**${visitId}**/sessions
- GET /visits/**${visitId}**
- GET /visits/**${visitId}**/events
- GET /visits/**${visitId}**/identities
- GET /visits/**${visitId}**/pages
- GET /visits/**${visitId}**/sessions

Event Resource

	Describes the event resource.

Description

The event resource contains the information relative to a business or system event that occurred on a specific web page at a given time. Two types of event can be issued:

- **SYSTEM** events are constant and cannot be customized. There are two groups of System events:
  - Visit related (VisitStarted, PageEntered, PageExited);
  - Identity related (SignIn, SignOut, UserInfo).
- **BUSINESS** events are custom events that you can define within the Browser DSL.

The possible event names are listed in the following table:

Name	Type	Description
VisitStarted	SYSTEM	Generated by the browser tier when the customer starts visiting the web site and enters in a first page. It creates a new visit resource. Then, all the pages visited by the customer are associated with this visit resource.
PageEntered	SYSTEM	Generated by the browser tier when the customer enters a page; the Backend Server creates a page resource which is associated with the visit. The page resource can be associated with an identity and a session, according to the identification scope of the customer.
PageExited	SYSTEM	Generated by the browser tier when the customer exits a page; the Backend Server updates accordingly the page resource. If the customer comes back on the page, a new page resource will be created.
SignIn	SYSTEM	Generated by the browser tier when the customer signs in (or

Name	Type	Description
		gets authenticated with the company web portal). It captures the identification information processed to authenticate the customer, for instance, the e-mail address used to login. When the Backend Server receives this event, it updates or creates the identity resource associated with the identity identifier. Additionally, it creates a new session with a unique `sessionID` across the visit, which becomes the new active session for the current visit.
SignOut	SYSTEM	Generated by the browser tier when the customer signs out. The end date of the session resource is updated and the session is no longer the visit's current active session.
UserInfo	SYSTEM	Generated when the browser tier submits information about the customer. This event occurs when the customer gets recognized or updates his or her profile information.
Timeout or InactivityTimeout	BUSINESS	Default business event with Timeout set to 10 seconds. Generated when the customer's mouse is no longer moving. This business event needs customization through the DSL rules. For further information, read Creating Business Events in the Developer's Guide.
Search	BUSINESS	Default business event. Generated when the web customer is searching on the web site. Needs further DSL customization. For further information, read Creating Business Events in the Developer's Guide section.
Custom name	BUSINESS	Custom business event created through DSL customization. For further information, read Creating Business Events in the Developer's Guide.

Resource Details

Field	Type	Mandatory	Description
eventID	string	yes	The unique ID<ref name="guid">GUID (Globally Unique Identifier)</ref> of the event resource.
eventName	string	yes	Name of the event. See the Description section for further information.
eventType	SYSTEM, BUSINESS	yes	Event type.
category	string	no	Category related to the generated event.
serverTimestamp	long	yes	Server timestamp.
browserPageID	string	yes	The browser page ID.
globalVisitID	string	yes	Global Visit ID<ref name="guid"/>.
url	string	yes	The url of the page.
timestamp	long	yes	Timestamp. (Difference with the Server timestamp?)
pageID	string	yes	ID<ref name="guid"/> of the associated page resource.
sessionID	string	no	ID<ref name="guid"/> of the associated session resource.
data	string[]	no	Additional JSON data, specific to the event.

<references />

Related Operations

You can retrieve event resources by using the following operations:

- Query event
- Query events by identity
- Query events by page
- Query events by session
- Create new event for visit
- Query events by visit

Examples

Retrieving a UserInfo Event

Operation

```
GET http://192.168.73.246:9081/backend/data/events/5cdca781-3fa3-11e2-aee5-00505625a04f
```

Response

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2012 15:48:08 GMT
Content-Type: application/json; charset=UTF-8
Date: Wed, 12 Dec 2012 15:48:08 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Content-Length: 587
{"eventID":"5cdca781-3fa3-11e2-aee5-00505625a04f",
"category":"",
"eventType":"SYSTEM","eventName":"UserInfo",
"serverTimestamp":1354798164472,"browserPageID":"2D869014426A4CAA8FA5C0D7B8668D0A",
"globalVisitID":"c93a19a1-45db-4d59-9c85-b637daea4a20",
"url":"http://www.genesyslab.com/
afu_FLS_intermediary.page?returnUrl=/?","timestamp":1354798163915,
"visitID":"f24c60f6-0728-4f3d-
b8b4-1e7bad2dc8a3","pageID":"5c911f90-3fa3-11e2-aee5-00505625a04f",
"data":{"userID":"user@genesyslab.com","sex":"male","name":"user1","age":30}}
```

Retrieving a PageEntered Event

Request

```
GET http://192.168.73.246:9081/backend/data/events/c4203381-3fa3-11e2-aee5-00505625a04f
```

Response

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2012 15:48:08 GMT
Content-Type: application/json; charset=UTF-8
Date: Wed, 12 Dec 2012 15:48:08 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Content-Length: 587
{"eventID":"c4203381-3fa3-11e2-aee5-00505625a04f","category":"",
"eventType":"SYSTEM","eventName":"PageEntered",
"serverTimestamp":1354798337720,"browserPageID":"AA8C3E0B8C9543D58D9CAB122A714124",
"globalVisitID":"c93a19a1-45db-4d59-9c85-b637daea4a20",
"url":"http://www.genesyslab.com/%3f","timestamp":1354798337628,
"visitID":"f24c60f6-0728-4f3d-b8b4-1e7bad2dc8a3",
"pageID":"c489cac0-3fa3-11e2-aee5-00505625a04f",
"data":{"urlReferrer":"http://www.genesyslab.com/afu_FLS_intermediary.page?returnUrl=/%3f",
"localTime":"2012-12-06T12:52:17.628Z",
"title":"404 - Not found"}}
```

Query event

	**GET /events/${eventId}**

Description

Retrieves a given event.

Operation

Method	GET		
URL	/events/**${eventId}**		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${eventId}	string	yes	Event identifier.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	Event

Example

The following operation retrieves a UserInfo event by its ID. Operation

`http://192.168.73.246:9081/backend/data/events/5cdca781-3fa3-11e2-aee5-00505625a04f`

Response

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2012 15:48:08 GMT
Content-Type: application/json; charset=UTF-8
Date: Wed, 12 Dec 2012 15:48:08 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Content-Length: 587
{"eventID":"5cdca781-3fa3-11e2-aee5-00505625a04f",
"category":"",
"eventType":"SYSTEM","eventName":"UserInfo",
"serverTimestamp":1354798164472,"browserPageID":"2D869014426A4CAA8FA5C0D7B8668D0A",
"globalVisitID":"c93a19a1-45db-4d59-9c85-b637daea4a20",
"url":"http://www.genesyslab.com/
afu_FLS_intermediary.page?returnUrl=/?","timestamp":1354798163915,
"visitID":"f24c60f6-0728-4f3d-
b8b4-1e7bad2dc8a3","pageID":"5c911f90-3fa3-11e2-aee5-00505625a04f",
"data":{"userID":"user@genesyslab.com","sex":"male","name":"user1","age":30}}
```

Identity Resource

	Describes the identity resource.

Description

The identity resource contains the customer information and is created when the Backend Server receives the SignIn event. The browser tier generates this event when the customer signs in (or get authenticated with the company web portal). The identification information submitted by the customer is used to create the identity ID (for instance, teh e-mail address or an account name). Additionally, it creates a new session with a unique session ID across the visit, which becomes the new active session for the current visit. At this point, the identification scope of the customer is `Authenticated`. If the customer signs out, the Backend Server receives the SignOut event and the identification scope becomes `Recognized`. For further information on events, see Event Resource.

Resource Details

Field	Type	Mandatory	Description
identityId	string	yes	The unique ID of the given identity. The user provides the identityId information when registering on the website; for instance, it can be the e-mail address.
name	string	no	User name.
location	string	no	User location.
entityInCS	string	no	Entity in the Contact Server. (TBC -> Contact ID?)
visitScope	Authenticated, Recognized	yes	Specify the visit's identification scope of the user. If the user is authenticated, you can retrieve session information. If the user signs out, the visit scope changes to recognized.
eventIds	string[]	no	Array of event IDs<ref name="guid">GUID

Field	Type	Mandatory	Description
			(Globally Unique Identifier)</ref> associated with this identity.
events	event[]	no	Array of the event resources associated with this identity.
pageIds	string[]	no	Array of page IDs<ref name="guid"/> associated with this identity.
pages	page[]	no	Array of the page resources associated with this identity.
sessionIds	string[]	no	Array of session IDs<ref name="guid"/> associated with this identity.
sessions	session[]	no	Array of the session resources associated with this identity.
visitIds	string[]	no	Array of visit IDs<ref name="guid"/> associated with this identity.
visits	visit[]	no	Array of the visit resources associated with this identity.

<references />

Related Operations

- Create new identity
- Create new visit for identity
- Query events by identity
- Query identity
- Query identities
- Query pages by identity
- Query sessions by identity
- Query visits by identity
- Query identities by visit

Examples

Retrieving Identities

The following operation retrieves all the identities available. Request

```
GET http://192.168.73.246:9081/backend/data/identities
```

Response

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2012 15:17:09 GMT
Content-Type: application/json; charset=UTF-8
Date: Wed, 12 Dec 2012 15:17:09 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Content-Length: 477
[{"eventIds":null,"events":null,"pageIds":null,"pages":null,
"sessionIds":null,"sessions":null,"identityId":"pat.thompsom@genesyslab.com",
"name":null,"location":null,"entityInCS":null,"visitScope":"Authenticated",
"visitIds":null,"visits":null},
{"eventIds":null,"events":null,"pageIds":null,"pages":null,
"sessionIds":null,"sessions":null,"identityId":"user@genesyslab.com",
"name":null,"location":null,"entityInCS":null,"visitScope":"Authenticated",
"visitIds":null,"visits":null}]
```

Retrieving Identities with page IDs

The following operation retrieves all the identities available, including the associated page IDs. Request

```
GET /backend/data/identities?include_pages=true
```

Response

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2012 15:28:13 GMT
Content-Type: application/json; charset=UTF-8
Date: Wed, 12 Dec 2012 15:28:13 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Content-Length: 627
[{"eventIds":null,"events":null,
"pageIds":["11ff6de0-446e-11e2-bfd2-00505625a04f"],
"pages":null,"sessionIds":null,"sessions":null,"
identityId":"pat.thompsom@genesyslab.com",
"name":null,"location":null,"entityInCS":null,
"visitScope":"Authenticated","visitIds":null,
"visits":null},
{"eventIds":null,"events":null,
"pageIds":["45f0eda1-3fa4-11e2-aee5-00505625a04f","c489cac0-3fa3-11e2-aee5-00505625a04f",
"06afd7f0-3fa4-11e2-aee50505625a04f"],"pages":null,"sessionIds":null,
"sessions":null,"identityId":"user@genesyslab.com",
"name":null,"location":null,"entityInCS":null,"visitScope":"Authenticated",
"visitIds":null,"visits":null}]
```

Create new identity

	**POST /identities**

Description

Creates a new identity.

Operation

Query identities

Method	POST
URL	/identities
BODY	Identity JSON object

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

Response

HTTP code	200
HTTP Title	OK

Create new visit for identity

	**POST /identities/${identityId}/visits**

Description

Creates a new visit for a given identity.

Operation

Query identities

Method	POST		
URL	/identities/${identityId}/visits		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${identityId}	string	yes	Identity identifier.
BODY	Visit JSON object		

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

Response

HTTP code	200
HTTP Title	OK

Query events by identity

	**GET /identities/${identityId}/events**

Description

Retrieves the events associated with a given identity.

Operation

Method	GET		
URL	/identities/**${identityId}**/events		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${identityId}	string	yes	Identity identifier.
age	integer	no	Events' maximum age in seconds. Older events will not be returned.
eventName	string	no	Event name.
eventType	string	no	Event type.
category	string or "all categories"	no	Category name or the "all categories" key, which means that the results include any event associated with a category or a combination of categories.
url	string	no	Event URL.
globalVisitID	string	no	Associated global Visit ID<ref name="guid">GUID (Globally Unique Identifier)</ref>.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Event resources.

Example

Activating Paging

The following example retrieves the first three events for the 'user@genesyslab.com' Identity. Operation

```
GET http://192.168.73.246:9081/backend/data/identities/user@genesyslab.com/events?page_size=3
```

Response

```
HTTP/1.1 200 OK
Date: Thu, 13 Dec 2012 14:26:47 GMT
Content-Type: application/json; charset=UTF-8
Date: Thu, 13 Dec 2012 14:26:47 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Paging-Next: user@genesyslab.com#1354798334842#c2994560-3fa3-11e2-aee5-00505625a04f
Content-Length: 1508
[{"eventID":"5cdca781-3fa3-11e2-aee5-00505625a04f",
"category":"","eventType":"SYSTEM","eventName":"UserInfo",
"serverTimestamp":1354798164472,
"browserPageID":"2D869014426A4CAA8FA5C0D7B8668D0A",
"globalVisitID":"c93a19a1-45db-4d59-9c85-b637daea4a20",
"url":"http://www.genesyslab.com/afu_FLS_intermediary.page?returnUrl=/?",
"timestamp":1354798163915,"visitID":"f24c60f6-0728-4f3d-b8b4-1e7bad2dc8a3",
"pageID":"5c911f90-3fa3-11e2-aee5-00505625a04f",
"data":{"userID":"user@genesyslab.com","sex":"male","name":"user1","age":30}},
{"eventID":"62a62150-3fa3-11e2-aee5-00505625a04f",
"category":"","eventType":"BUSINESS","eventName":"Timeout-10","serverTimestamp":1354798174181,
"browserPageID":"2D869014426A4CAA8FA5C0D7B8668D0A","globalVisitID":"c93a19a1-45db-4d59-9c85-b637daea4a20","url"
afu_FLS_intermediary.page?returnUrl=/%3f",
"timestamp":1354798174090,"visitID":"f24c60f6-0728-4f3d-b8b4-1e7bad2dc8a3",
"pageID":"5c911f90-3fa3-11e2-aee5-00505625a04f","data":{}},
{"eventID":"6e920a60-3fa3-11e2-aee5-00505625a04f","category":"",
"eventType":"BUSINESS","eventName":"Timeout-30","serverTimestamp":1354798194182,
"browserPageID":"2D869014426A4CAA8FA5C0D7B8668D0A",
"globalVisitID":"c93a19a1-45db-4d59-9c85-b637daea4a20",
"url":"http://www.genesyslab.com/afu_FLS_intermediary.page?returnUrl=/%3f",
"timestamp":1354798194128,"visitID":"f24c60f6-0728-4f3d-b8b4-1e7bad2dc8a3",
"pageID":"5c911f90-3fa3-11e2-aee5-00505625a04f","data":{}}]
```

Retrieving the next page of results

The following request uses the Paging-Next header parameter of the previous response (`user@genesyslab.com#1354798334842#c2994560-3fa3-11e2-aee5-00505625a04f`) to retrieve the next three events. Request

```
GET http://192.168.73.246:9081/backend/data/identities/user@genesyslab.com/events?page_size=3
&page_value="user@genesyslab.com#1354798334842#c2994560-3fa3-11e2-aee5-00505625a04f"≠xt=true
```

Response

```
HTTP/1.1 200 OK
Date: Thu, 13 Dec 2012 14:35:11 GMT
Content-Type: application/json; charset=UTF-8
Date: Thu, 13 Dec 2012 14:35:11 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Paging-Next: pat.thompsom@genesyslab.com#1355325030949#11ed4570-446e-11e2-bfd2-00505625a04f
Paging-Prev: "user@genesyslab.com10135479833484210c2994560-3fa3-11e2-aee5-00505625a04f"
Content-Length: 1507
[{"eventID":"03956571-446e-11e2-bfd2-00505625a04f","category":"Internet","eventType":"SYSTEM","eventName":"Sign
"serverTimestamp":1355325007175,"browserPageID":"7FA47396264C4F6A8FED4F4E9710B0C4",
"globalVisitID":"66c0976d-cbdd-4e94-accb-26093803cf54",
"url":"http://www.genesyslab.com/catalogue-topic/t_new_internet_options.page?",
"timestamp":1355325006715,
"visitID":"42788a69-785d-4860-be39-45048d32441c",
"pageID":"f57b03a0-446d-11e2-bfd2-00505625a04f",
"data":{"userID":"pat.thompsom@genesyslab.com"}},
{"eventID":"072fa330-446e-11e2-bfd2-00505625a04f",
"category":"Internet",
"eventType":"BUSINESS","eventName":"Timeout-30",
"serverTimestamp":1355325013219,
"browserPageID":"7FA47396264C4F6A8FED4F4E9710B0C4",
"globalVisitID":"66c0976d-cbdd-4e94-accb-26093803cf54",
"url":"http://www.genesyslab.com/catalogue-topic/t_new_internet_options.page?",
"timestamp":1355325013148,
"visitID":"42788a69-785d-4860-be39-45048d32441c",
"pageID":"f57b03a0-446d-11e2-bfd2-00505625a04f",
"data":{}},
{"eventID":"0fbe5780-446e-11e2-bfd2-00505625a04f",
"category":"Internet","eventType":"SYSTEM",
"eventName":"PageExited","serverTimestamp":1355325027576,
"browserPageID":"7FA47396264C4F6A8FED4F4E9710B0C4",
"globalVisitID":"66c0976d-cbdd-4e94-accb-26093803cf54",
"url":"http://www.genesyslab.com/catalogue-topic/t_new_internet_options.page?",
"timestamp":1355325027495,
"visitID":"42788a69-785d-4860-be39-45048d32441c",
"pageID":"f57b03a0-446d-11e2-bfd2-00505625a04f",
"data":{}}]
```

Query identity

	**GET /identities/${identityId}**

Description

Retrieves a given identity.

Operation

Method	GET		
URL	/identities/**${identityId}**		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${identityId}	string	yes	Identity identifier.
include_visits	• `true` • `false`	no	If `true`, the returned identity contains the reference list of associated visits.
include_visits_detail	• `true` (if `include_visits=true`) • `false`	no	If `true`, the returned identity contains the associated visit resources. You can only use this parameter if `include_visits` is set to `true`.
include_sessions	• `true` • `false`	no	If `true`, the returned identity contains the reference list of authentication sessions.
include_sessions_detail	• `true` (if `include_sessions=true`) • `false`	no	If `true`, the returned identity contains the associated sessions. You can only use this parameter if `include_sessions` is set to `true`.
include_pages	• `true`	no	If `true`, the returned identity contains the reference list of the

Method	GET		
	• `false`		associated pages.
include_pages_detail	• `true` (if `include_pages=true`) • `false`	no	If `true`, the returned identity contains the associated pages. You can only use this parameter if `include_pages` is set to `true`.
include_events	• `true` • `false`	no	If `true`, the returned identity contains the reference list of associated events.
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, the returned identity contains the associated events. You can only use this parameter if `include_events` is set to `true`.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Identity.

Examples

The following operation retrieves the identity resource with pat.thompsom@genesyslab.com ID.
Request

```
http://192.168.73.246:9081/backend/data/identities/pat.thompsom@genesyslab.com
```

Response

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2012 15:17:09 GMT
Content-Type: application/json; charset=UTF-8
```

```
Date: Wed, 12 Dec 2012 15:17:09 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Content-Length: 477
[{"eventIds":null,"events":null,"pageIds":null,"pages":null,
"sessionIds":null,"sessions":null,"identityId":"pat.thompsom@genesyslab.com",
"name":null,"location":null,"entityInCS":null,"visitScope":"Authenticated",
"visitIds":null,"visits":null}]
```

Query identities

	**GET /identities**

Description

Retrieves a list of identities. By default, only the identities are returned: the associated visits, sessions, events, and pages are not included in the results.

Operation

Method	GET		
URL	/identities		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
location	string	no	Filters the identity results based on the given geolocation parameter.
userAgent	string	no	Returns identities with an exact match for the UserAgent attribute.
include_visits	• `true` • `false`	no	If `true`, each returned identity contains the reference list of associated visits.
include_visits_detail	• `true` (if `include_visits=true`) • `false`	no	If `true`, each returned identity contains the associated visits resources. You can only use this parameter if `include_visits` is set to `true`.
include_sessions	• `true` • `false`	no	If `true`, each returned identity contains the reference list of authentication sessions.

Method	GET		
include_sessions_detail	• `true` (if `include_sessions=true`) • `false`	no	If `true`, each returned identity contains the associated authentication sessions. You can only use this parameter if `include_sessions` is set to `true`.
include_pages	• `true` • `false`	no	If `true`, each returned identity contains the reference list of the associated pages.
include_pages_detail	• `true` (if `include_pages=true`) • `false`	no	If `true`, each returned identity contains the associated pages. You can only use this parameter if `include_pages` is set to `true`.
include_events	• `true` • `false`	no	If `true`, each returned identity contains the reference list of associated events.
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, each returned identity contains the associated events.You can only use this parameter if `include_events` is set to `true`.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

Response	
HTTP code	200
HTTP Title	OK
Body	JSON array of Identity.

Example

Retrieving Identities with page IDs

The following operation retrieves all the identities available, including the associated page IDs.
Request

```
GET /backend/data/identities?include_pages=true
```

Response

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2012 15:28:13 GMT
Content-Type: application/json; charset=UTF-8
Date: Wed, 12 Dec 2012 15:28:13 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Content-Length: 627
[{"eventIds":null,"events":null,
"pageIds":["11ff6de0-446e-11e2-bfd2-00505625a04f"],
"pages":null,"sessionIds":null,"sessions":null,"
identityId":"pat.thompsom@genesyslab.com",
"name":null,"location":null,"entityInCS":null,
"visitScope":"Authenticated","visitIds":null,
"visits":null},
{"eventIds":null,"events":null,
"pageIds":["45f0eda1-3fa4-11e2-aee5-00505625a04f","c489cac0-3fa3-11e2-aee5-00505625a04f",
"06afd7f0-3fa4-11e2-aee50505625a04f"],"pages":null,"sessionIds":null,
"sessions":null,"identityId":"user@genesyslab.com",
"name":null,"location":null,"entityInCS":null,"visitScope":"Authenticated",
"visitIds":null,"visits":null}]
```

Query pages by identity

	**GET /identities/${identityId}/pages**

Description

Retrieves the pages for a given identity.

Operation

Method	GET		
URL	/identities/**${identityId}**/pages		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${identityId}	string	yes	Identity identifier.
age	integer	no	Pages' maximum age in seconds. Older pages will not be returned.
include_events	• `true` • `false`	no	If `true`, the returned pages contain the reference list of associated events.
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, the returned pages contain the associated events. You can only use this parameter if `include_events` is set to `true`.
url	string	no	Page URL used to filter the results.
title	string	no	Page title used to filter the results.
category	string or "all categories"	no	A specific category name or the "all categories" key, which means that the results include any page associated with a category or a

Method	GET		
			combination of categories.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Page resources.

Query sessions by identity

	**GET /identities/${identityId}/sessions**

Description

Retrieves the sessions for a given identity.

Operation

Method	GET		
URL	/identities/**${identityId}**/sessions		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${identityId}	string	yes	Identity identifier.
age	integer	no	Sessions' maximum age in seconds. Older sessions will not be returned.
include_pages	• `true` • `false`	no	If `true`, the returned sessions contain the reference list of the associated pages.
include_pages_detail	• `true` (if `include_pages=true`) • `false`	no	If `true`, the returned sessions contain the associated pages.You can only use this parameter if `include_pages` is set to `true`.
include_events	• `true` • `false`	no	If `true`, the returned sessions contain the reference list of associated events.
include_events_detail	• `true` (if `include_events=true`)	no	If `true`, the returned sessions contain the associated events. You can only use this parameter if `include_events` is set

Method	GET		
	• `false`		to `true`.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Session resources.

Query visits by identity

	**GET /identities/${identityId}/visits**

Description

Retrieves the visits for a given identity.

Operation

Method	GET		
URL	/identities/**${identityId}**/visits		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${identityId}	string	yes	Identity identifier.
age	integer	no	Visits' maximum age in seconds. Older visits will not be returned.
globalVisitID	string	no	Filters the visit results based on the given global Visit ID.
userAgent	string	no	Returns the visits matching the UserAgent field.
include_sessions	• `true` • `false`	no	If `true`, the returned visits contain the reference list of sessions.
include_sessions_detail	• `true` (if `include_sessions=true`) • `false`	no	If `true`, the returned visits contain the associated sessions. You can only use this parameter if `include_sessions` is set to `true`.
include_pages	• `true`	no	If `true`, the returned visits contain the reference list of the

Method	GET		
	• `false`		associated pages.
include_pages_detail	• `true`(if `include_pages=true`) • `false`	no	If `true`, the returned visits contain the associated pages. You can only use this parameter if `include_pages` is set to `true`.
include_events	• `true` • `false`	no	If `true`, the returned visits contain the reference list of associated events.
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, the returned visits contain the associated events. You can only use this parameter if `include_events` is set to `true`.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Visit resources.

Page Resource

	Defines the page resource.

Description

Defines a page resource. The page resource is created when the customer enters a page (upon the `PageEntered` event). If the customer leaves the page, and later visits the page again, a new page resource is created. For further information about events, see Event Resource.

Resource Details

Field	Type	Mandatory	Description
pageID	string	yes	The unique ID of the page resource.
url	string	yes	The page URL.
browserPageID	string	yes	The browser page ID; another page identifier unique across the visit.
pageExitedDate	long	no	Date at which the user left the page. 0 means that the user did not leave the page already.
pageEnteredDate	long	no	Date at which the user entered the page.
category	string	no	Associated category, if any.
title	string	no	Page title.
first	• `true` • `false`	no	`true` if this is the first page entered for the current visit.
eventIds	string[]	no	IDs of the associated events.
events	event[]	no	Array of the associated events.

<references />

Related Operations

- Create new event for page
- Query events by page
- Query page
- Query pages by identity
- Query pages by session
- Create new page for visit
- Query pages by visit

Example

Retrieving a page

Operation

```
GET http://192.168.73.246:9081/backend/data/pages/c489cac0-3fa3-11e2-aee5-00505625a04f
```

Response


```
{"eventIds":null,"events":null,"pageId":"c489cac0-3fa3-11e2-aee5-00505625a04f",
"url":"http://www.genesyslab.com/%3f",
"browserPageID":"AA8C3E0B8C9543D58D9CAB122A714124",
"pageExitedDate":1354798339992,
"pageEnteredDate":1354798338412,
"category":"","title":"404 - Not found","first":false}
```

Create new event for page

	**POST /pages/${pageId}/events**

Description

Creates a new event for a given page resource.

Operation

Query identities

Method	POST		
URL	/pages/**${pageId}**/events		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${pageId}	string	yes	Page identifier.
BODY	Event JSON object		

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

Response

HTTP code	200
HTTP Title	OK

Query events by page

	**GET /pages/${pageId}/events**

Description

Retrieves the events for a given page.

Operation

Method	GET		
URL	/pages/**${pageId}**/events		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${pageId}	string	yes	Page identifier.
age	integer	no	Events' maximum age in seconds. Older events will not be returned.
eventName	string	no	Event name.
eventType	string	no	Event type.
category	string or "all categories"	no	Category name or the "all categories" key, which means that the results include any event associated with a category or a combination of categories.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Event resources

Query page

	**GET /pages/${pageId}**

Description

Retrieves a given page.

Operation

Method	GET		
URL	/pages/**${pageId}**		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${pageId}	string	yes	Page identifier.
include_events	• `true` • `false`	no	If `true`, the returned page contains the reference list of associated events.
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, the returned page contains the associated events. You can only use this parameter if `include_events` is set to `true`.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK

HTTP code	200
Body	Page

Session Resource

	Defines the session resource.

Description

The session resource contains the session information relative to a customer visit after a SignIn event was received. The session resource is created when the customer authenticates on the website. Then, all the events and pages involved in the customer's browsing activity are associated with the session till the session ends (the user signs out). For further information about events, see Event Resource.

Resource Details

Field	Type	Mandatory	Description
sessionID	string	yes	The unique ID of the session.
identityId	string	yes	The ID of the associated Identity resource.
startDate	long	yes	The date at which the session started (when the customer signed in).
endDate	long	no	The date at which the session ended (when the customer signed out). 0 means that the session is not terminated.
duration	int	no	The session duration, in seconds.
eventIds	string[]	no	Array of event IDs; all the events which occurred during the session.
events	event[]	no	Array of the associated events.
pageIds	string[]	no	Array of page IDs; all the pages visited during the current session.
pages	page[]	no	Array of the pages visited during this

Field	Type	Mandatory	Description
			session.

<references />

Related Operations

- Create new page for session
- Query events by session
- Query pages by session
- Query session
- Query sessions by visit
- Create new session for visit
- Query sessions by identity
- Query visits by identity

Example

The following request retrieves a session resource with its session ID. See Query session for further information. Request

```
GET http://192.168.73.246:9081/backend/data/sessions/461d54d0-3fa4-11e2-aee5-00505625a04f
```

Response

```
 HTTP/1.1 200 OK
Date: Wed, 12 Dec 2012 16:20:11 GMT
Content-Type: application/json; charset=UTF-8
Date: Wed, 12 Dec 2012 16:20:11 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Content-Length: 208
{"eventIds":null,"events":null,"pageIds":null,"pages":null,
"sessionId":"461d54d0-3fa4-11e2-aee5-00505625a04f",
"startDate":1354798555805,"endDate":1354798556259,
"duration":0,"identityId":"user@genesyslab.com"}
```

Create new page for session

	**POST /sessions/${sessionId}/pages**

Description

Creates a new page for a given session resource.

Operation

Query identities

Method	POST		
URL	/sessions/${sessionId}/pages		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${sessionId}	string	yes	Session identifier.
BODY	Page JSON object		

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

Response

HTTP code	200
HTTP Title	OK

Query events by session

	**GET /sessions/${sessionId}/events**

Description

Retrieves the events for a given session.

Operation

Method	GET		
URL	/sessions/**${sessionId}**/events		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${sessionId}	string	yes	Session identifier.
age	integer	no	Events' maximum age in seconds. Older events will not be returned.
eventName	string	no	Event name.
eventType	string	no	Event type.
category	string or "all categories"	no	Category name or the "all categories" key, which means that the results include any event associated with a category or a combination of categories.
url	string	no	Event URL.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Event resources

Query pages by session

	**GET /sessions/${sessionId}/pages**

Description

Retrieves the pages for a given session.

Operation

Method	GET		
URL	/sessions/**${sessionId}**/pages		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${sessionId}	string	yes	Session identifier.
age	integer	no	Pages' maximum age in seconds. Older pages will not be returned.
include_events	• `true` • `false`	no	If `true`, the returned pages contain the reference list of associated events.
include_events_detail	• `true` (if `include_events="true"`) • `false`	no	If `true`, the returned pages contain the associated events. You can only use this parameter if `include_events` is set to `true`.
url	string	no	Page URL.
title	string	no	Page title.
category	string or "all categories"	no	A specific category name or the "all categories" key, which means that the results include any page associated with a category or a combination of categories.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Page resources.

Query session

	**GET /sessions/${sessionId}**

Description

Retrieves a specific session resource.

Operation

Method	GET		
URL	/sessions/**${sessionId}**		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${sessionId}	string	yes	Authenticated session's identifier.
include_pages	• `true` • `false`	no	If `true`, the returned session contains the reference list of the associated pages.
include_pages_detail	• `true`(if `include_pages=true`) • `false`	no	If `true`, the returned session contains the associated pages.You can only use this parameter if `include_pages` is set to `true`.
include_events	• `true` • `false`	no	If `true`, the returned session contains the reference list of associated events.
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, the returned session contains the associated events. You can only use this parameter if `include_events` is set to `true`.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	Session

Visit Resource

	Describes the Visit Resource.

Description

The visit resource contains the information related to a customer's visit, which is the browser session opened by the customer to visit the website. The visit starts with the `VisitStarted` event (submitted by the website) and then, all the browsing activity is recorded in page, session, and event resources associated with the visit. The customer remains `Anonymous` till he or she authenticates on the website. When the authentication occurs, the visit is associated with an identity resource, in addition to the new session resource created. Note that several identities will be associated with the same visitId if several customers authenticate on the website. See Identity Resource.

Resource Details

Field	Type	Mandatory	Description
visitId	string	yes	ID<ref name="guid">GUID (Globally Unique Identifier)</ref> of the given visit.
startDate	long	yes	Date in milliseconds at which the visit started.
endDate	long	no	Date in milliseconds at which the visit ended; 0 means that the visit is not terminated.
activeSessionId	string	no	ID<ref name="guid"/> of the current active session if any; otherwise, null.
globalVisitID	string	no	ID<ref name="guid"/> of the global visit.
userAgentID	string	no	ID<ref name="guid"/> of the associated user agent if any; otherwise, null.
eventIds	string[]	no	Array of event IDs<ref name="guid"/> associated with this visit.

Field	Type	Mandatory	Description
events	event[]	no	Array of the event resources associated with this visit.
pageIds	string[]	no	Array of page IDs<ref name="guid"/> associated with this visit.
pages	page[]	no	Array of the page resources associated with this visit.
sessionIds	string[]	no	Array of session IDs<ref name="guid"/> associated with this visit.
sessions	session[]	no	Array of the session resources associated with this visit.

<references />

Related Operations

- Create new event for visit
- Create new page for visit
- Create new session for visit
- Create new visit
- Query events by visit
- Query identities by visit
- Query pages by visit
- Query sessions by visit
- Query visit
- Create new visit for identity
- Query visits by identity

Example

Retrieve a Given Visit

Request

```
GET http://192.168.73.246:9081/backend/data/visits/f24c60f6-0728-4f3d-b8b4-1e7bad2dc8a3
```

Response

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2012 16:03:03 GMT
Content-Type: application/json; charset=UTF-8
Date: Wed, 12 Dec 2012 16:03:03 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.0
Content-Length: 345
{"eventIds":null,"events":null,"pageIds":null,"pages":null,"
sessionIds":null,"sessions":null,
"visitId":"f24c60f6-0728-4f3d-b8b4-1e7bad2dc8a3",
"startDate":1354797882714,"endDate":0,
"activeSessionId":"00000000-0000-1000-8000-000000000000",
"globalVisitID":"c93a19a1-45db-4d59-9c85-b637daea4a20",
"userAgentId":"b4ec8ef0-3fa2-11e2-aee5-00505625a04f"}
```

Create new visit

	**POST /visits**

Description

Creates a new visit resource.

Operation

Query identities

Method	POST
URL	/visits
BODY	Visit JSON object

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

Response

HTTP code	200
HTTP Title	OK

Create new event for visit

	**POST /visits/${visitId}/events**

Description

Creates a new event for a given visit resource.

Operation

Query identities

Method	POST		
URL	/visits/**${visitId}**/events		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${visitId}	string	yes	Visit identifier.
BODY	Event JSON object which MUST contains the pageId field, to ensure that the parent page is defined for this new event.		

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

Response

HTTP code	200
HTTP Title	OK

Create new session for visit

	**POST /visits/${visitId}/sessions**

Description

Creates a new session for a given visit resource.

Operation

Query identities

Method	POST		
URL	/visits/${visitId}/sessions		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${visitId}	string	yes	Visit identifier.
BODY	Session JSON object		

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

Response

HTTP code	200
HTTP Title	OK

Create new page for visit

	**POST /visits/${visitId}/pages**

Description

Creates a new page for a given visit resource.

Operation

Query identities

Method	POST		
URL	/visits/**${visitId}**/pages		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${visitId}	string	yes	Visit identifier.
BODY	Page JSON object		

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

Response

HTTP code	200
HTTP Title	OK

Query identities by visit

	GET **/visits/${visitId}/identities**

Description

Retrieves a list of identities for the given visit ID.

Operation

Method	GET		
URL	/visits/${visitId}/identities		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
include_visits	• `true` • `false`	no	If `true`, each returned identity contains the reference list of associated visits.
include_visits_detail	• `true` (if `include_visits=true`) • `false`	no	If `true`, each returned identity contains the associated visits resources. You can only use this parameter if `include_visits` is set to `true`.
include_sessions	• `true` • `false`	no	If `true`, each returned identity contains the reference list of sessions.
include_sessions_detail	• `true` (if `include_sessions=true`) • `false`	no	If `true`, each returned identity contains the associated sessions.You can only use this parameter if `include_sessions` is set to `true`.
include_pages	• `true`	no	If `true`, each returned identity contains the reference list of the

Method	GET		
	• `false`		associated pages.
include_pages_detail	• `true` (if `include_pages=true`) • `false`	no	If `true`, each returned identity contains the associated pages. You can only use this parameter if `include_pages` is set to `true`.
include_events	• `true` • `false`	no	If `true`, each returned identity contains the reference list of associated events.
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, each returned identity contains the associated events. You can only use this parameter if `include_events` is set to `true`.
association	• "Authenticated" • "Recognized"	no	Defines the type of association between the current visit and the returned identity.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON array of Identity

Query sessions by visit

	**GET /visits/${visitId}/sessions**

Description

Retrieves the sessions for a given visit.

Operation

Method	GET		
URL	/visits/**${visitId}**/sessions		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${visitId}	string	yes	visit identifier.
identityId	string	no	ID of the identity associated with the returned sessions.
age	integer	no	Sessions' maximum age in seconds. Older sessions will not be returned.
include_pages	• `true` • `false`	no	If `true`, the returned sessions contain the reference list of the associated pages.
include_pages_detail	• `true`(if `include_pages=true`) • `false`	no	If `true`, the returned sessions contain the associated pages. You can only use this parameter if `include_pages` is set to `true`.
include_events	• `true` • `false`	no	If `true`, the returned sessions contain the reference list of associated events.

Method	GET		
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, the returned sessions contain the associated events. You can only use this parameter if `include_events` is set to `true`.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Session resources.

Query pages by visit

	**GET /visits/${visitId}/pages**

Description

Retrieves the pages for a given visit.

Operation

Method	GET		
URL	/visits/**${visitId}**/pages		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${visitId}	string	yes	Visit identifier.
age	integer	no	Pages' maximum age in seconds. Older pages will not be returned.
include_events	• `true` • `false`	no	If `true`, the returned pages contain the reference list of associated events.
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, the returned pages contain the associated events. You can only use this parameter if `include_events` is set to `true`.
url	string	no	Page URL used to filter the results.
title	string	no	Page title used to filter the results.
category	string or "all categories"	no	A specific category name or the "all categories" key, which means that the results include any page associated with a category or a

Method	GET		
			combination of categories.
browserPageID	string	no	Browser page id, which is another page identifier unique across the visit. A collection with maximum one page will be returned.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Page resources.

Query events by visit

	**GET /visits/${visitId}/events**

Description

Retrieves the events for a given visit.

Operation

Method	GET		
URL	/visits/**${visitId}**/events		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${visitId}	string	yes	Visit identifier.
age	integer	no	Events' maximum age in seconds. Older events will not be returned.
eventName	string	no	Event name.
eventType	string	no	Event type.
category	string or "all categories"	no	Category name or the "all categories" key, which means that the results include any event associated with a category or a combination of categories.
url	string	no	Event URL.
browserPageID	string	no	Browser page ID of the event, which is another page identifier unique across the visit.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	JSON Array of Event resources.

Query visit

	**GET /visits/${visitId}**

Description

Retrieves a given visit.

Operation

Method	GET		
URL	/visits/**${visitId}**		
Name	**Value**	**Mandatory**	**Description**
URI Parameters			
${visitId}	string	yes	Visit identifier.
include_sessions	• `true` • `false`	no	If `true`, the returned visit contains the reference list of authentication sessions.
include_sessions_detail	• `true` (if `include_sessions=true`) • `false`	no	If `true`, the returned visit contains the associated authentication sessions. You can only use this parameter if `include_sessions` is set to `true`.
include_pages	• `true` • `false`	no	If `true`, the returned visit contains the reference list of the associated pages.
include_pages_detail	• `true` (if `include_pages=true`) • `false`	no	If `true`, the returned visit contains the associated pages. You can only use this parameter if `include_pages` is set to `true`.
include_events	• `true`	no	If `true`, the returned visit contains the reference list of

Method	GET		
	• `false`		associated events.
include_events_detail	• `true` (if `include_events=true`) • `false`	no	If `true`, the returned visit contains the associated events. You can only use this parameter if `include_events` is set to `true`.

<references />

Response

The History REST API answers with HTTP codes for every request. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP Title	OK
Body	Visit.