

1 Document

An XML file containing all project information to load and execute the project on the client. Certain interfaces are optional, however others are mandatory. This guide should reflect the changes in the XML project and keep track of the versions. Hopwfully this can remain simple!

2 Root

The XML root must be `<BrowserEvalProjectDocument>`. This should be sufficiently identifiable in both itself and in the JavaScript decoding as it will create an object called the root name.

There must also be a `<version>` tag which has the attribute `id` containing a numerical representation of the version. Currently everything in this document can be assumed to be version 1. If future updates or corrections are made post delivery this should give the flexibility to ensure past projects still work.

The root will also contain the following tags: setup and tracks.

3 Setup tag

The setup tag specifies certain global test settings including: the interface type to use, the project return location and any other setup instructions. Any general pre/post test questions must be specified in the relevant children tag. Any enabled metrics must also be specified in the metric child node.

3.1 Attributes

- **interface** - Mandatory, String. Defaults to APE, otherwise use to load any of the available interfaces. Currently only valid string is APE.
- **projectReturn** - Mandatory, String. Specify the URL to return the test results. If null client will generate XML locally and prompt user to return the file.
- **randomiseOrder** - Optional, default to false. Specify if the order of the tests can be randomised.
- **collectMetrics** - Optional, Boolean. Default to false. Determine if the test metrics should be collected. These include how long each test session took etc. The full metrics list can be modified in the 'metrics' tag.

3.2 Elements

None

4 AudioHolder tag

There should be one audioHolder tag per test session, inside which each audioElement is specified as children. The audioHolder tag can help to generalise certain objects. Each audioHolder instance specifies a separate listening test to be paged, each with their own specific requirements.

4.1 Attributes

- **id** - Mandatory, String. Give an ID string or number to identify the test in the result.
- **hostURL** - Optional, String. If all tracks are hosted from the same folder on a server, you can put in the lead here. For instance, if loading `http://test.com/tracks/track1.wav` and `http://test.com/tracks/track2.wav`, this could equal `http://test.com/tracks/` and the url attribute in the track tag can be `track1.wav` or `track2.wav`. Equally `http://test.com/` and then using `tracks/track1.wav` and `tracks/track2.wav` is valid.
- **sampleRate** - Optional, Number. If your test requires a specific sample rate, this should be set to the desired sample rate in Hertz. This does not set the browser to the correct sample rate, but forces the browser to check the sample rate matches. If this is undefined, no sample rate matching will occur.
- **randomiseOrder** - Optional, Boolean String. Defaults to false. Determine if the track order should be randomised. Must be true or false.
- **repeatCount** - Optional, Number. Defaults to 0 (ie: no repeats). The number of times a test should be repeated.

4.2 Elements

Contain the `audioElements` tags and the `interfaceSetup` tag.

5 audioElements tag

This must reside as children in the `audioHolder` tag. There must be one `audioElement` tag per sound sample to load into the test.

5.1 Attributes

- **id** - Mandatory, String. Must give a string or number to identify each audio element. This id is used in the output to identify each track once randomised.
- **url** - Mandatory, String. Contain the full URL to the track. If the `Tracks` tag `hostURL` is set, concatenate this tag with the `hostURL` attribute to obtain the full URL.

6 interface tag

This is contained within the `audioHolder` tag and outlines test instance specific requirements. These include the following children tags:

- **'title'** - Contains the test title to be shown at the top of the page. Can only be one title node per interface.
- **'scale'** - Takes the attribute position to be a value between 0 and 100 indicating where on the scale to place the text contained inside. Can be multiple scale tags per interface.

7 CommentQuestion tag

This is a 1st level tag (same level as AudioHolder and setup). This allows another question and comment box to be presented on the page. The results of these are passed back in the results XML with both the comment and the question. The id attribute is set to keep track at the results XML.

8 PreTest tag and PostTest tag

These are 1st level tags. The PreTest tag allows for the specifying of pre test instructions and questions. These appear as a pop-up style window with next buttons and other automatic GUI. The postTest tag allows for specifying post test instructions, questions and resources. These appear as a pop-up style window after the submit button is pressed.

8.1 Attributes

None.

8.2 Elements

Takes the `statement` and `question` tags. The order these are presented in the XML define the order they appear on the screen.

8.2.1 Statement

The statement tag simply prints the included string verbatim on a 'pop-up' window with a next button.

8.2.2 Question

This allows for a question to be asked pre/post the test. This is added to the response XML in the same location as the other common/global questions. The response includes both the question asked and the response. This takes two attributes, id and mandatory. ID is a mandatory field. The same ID will be used in the results so it is important it is properly entered. Mandatory is optional. True means the field must be entered before continuing.

8.2.3 Resource

The resource tag is only available in the postTest tag. This allows for the linking to some external resource via the href attribute.

9 Metric tag

A 1st level tag, metrics must be declared in the setup tag. This takes a set of children 'metricEnable' to define which metrics to collect and present.

9.1 metricEnable tag

This takes a single attribute to determine which metric to enable for collection. Some of these are a global, per track or per test instance.

- testTimer - Return the global test timer and test instance timers. Measures the time between the first start and final submit.
- elementTimer - Return the total time each audioElement in each test was listened too. Measures time between successive clicks on the track changer
- elementTracker - Return the initial position of each track
- elementTrackerFull - Return an enumerated pair of time and position. Track the entire movement of each element position. NOTE: Will override the elementTracker option above and throw an error into the browser console.
- elementFlagListenedTo - Return a boolean per element to see if the element was listened to
- elementFlagMoved - Return a boolean per element to see if the element slider was moved.
- elementFlagComments - Return a boolean per element to see if the element has comments.

10 Feature List

- Paging listening tests - eg. Ask multiple questions in each experiment
- Labels on X axis - scale
- Input questions/comment at top to guide towards the question being asked.
- Randomise track numbers -(inc. comment boxes and relate back to correct reference track)
- Randomise order of individual tests
- Save output XML file to remote server
- Tests Metrics
 - Duration of listening to each track
 - Time spent on each individual test
 - Start and end position of every track
 - Flags on each track, to ensure each track (but may not restrict users from submitting)
 - * Has been listened to
 - * Has been moved
 - * Has comments about it

10.1 Advanced feature list

- Repeat each tests number of times (2 or 3?) to remove learning / experience bias and ensure that the order is consistent
- Perform Loudness equalisation on all tracks
- Selection of test type
- Pre-test of some basic hearing test
 - MUSHRA (with vertical slider per track)
 - APE (Single horizontal slider)
 - AB Test

11 Example

Here is an example XML structure

```
<?xml version="1.0" encoding="utf-8"?>
<BrowserEvalProjectDocument>
  <setup interface="APE" projectReturn="null" randomiseOrder='true' collection="">
    <PreTest>
      <statement>Please listen to all mixes</statement>
      <question id="location" mandatory="true">Please enter your location</question>
    </PreTest>
    <PostTest>
      <statement>Thank you for taking this listening test.</statement>
      <question id="SessionID">Please enter your name.</question>
    </PostTest>
    <Metric>
      <metricEnable>testTimer</metricEnable>
      <metricEnable>elementTimer</metricEnable>
      <metricEnable>elementTracker</metricEnable>
      <metricEnable>elementFlagListenedTo</metricEnable>
      <metricEnable>elementFlagMoved</metricEnable>
    </Metric>
  </setup>
  <audioHolder id='0' hostURL="example_eval/" sampleRate="44100" randomiseOrder="true">
    <interface>
      <title>Example Test Question</title>
      <scale position="0">Min</scale>
      <scale position="100">Max</scale>
      <scale position="50">Middle</scale>
      <scale position="20">20</scale>
    </interface>
    <audioElements url="0.wav" id="0"/>
    <audioElements url="1.wav" id="1"/>
    <audioElements url="2.wav" id="2"/>
    <audioElements url="3.wav" id="3"/>
    <audioElements url="4.wav" id="4"/>
  </audioHolder>
</BrowserEvalProjectDocument>
```

```

<audioElements url="5.wav" id="5"/>
<audioElements url="6.wav" id="6"/>
<audioElements url="7.wav" id="7"/>
<audioElements url="8.wav" id="8"/>
<audioElements url="9.wav" id="9"/>
<audioElements url="10.wav" id="10"/>
<CommentQuestion id='mixingExperiance'>What is your mixing exper
<PreTest>
    <statement>Start the Test 3</statement>
</PreTest>
<PostTest>
    <statement>Please take a break before the next test</sta
    <question id="testComment">How did you find the test</q
</PostTest>
</audioHolder>
</BrowserEvalProjectDocument>

```