Web Browser page API

The Web Browser VR node shows a web page on a plane in the VR viewers. When you set its Source to HTML, you type the page in Composer, and the viewers load that text as the page. The viewers add a set of JavaScript functions to the page's window. A page from Source HTML always gets them. A page from a URL gets them in the VR Viewer; the web viewer adds them only to pages served from its own origin. These are the same functions that Panel Designer pages use. With them, a page can fire Training Builder events, play animations, apply scene states, and read and write variables.

- Web Browser page API

Web Browser page API

The Web Browser VR node shows a web page on a plane in the VR viewers. When you set its Source to HTML, you type the page in Composer, and the viewers load that text as the page. The viewers add a set of JavaScript functions to the page's `window`. A page from Source HTML always gets them. A page from a URL gets them in the VR Viewer; the web viewer adds them only to pages served from its own origin. These are the same functions that Panel Designer pages use. With them, a page can fire Training Builder events, play animations, apply scene states, and read and write variables.

Availability

The viewer dispatches the `simlabready` event on `window` once the functions exist. A page loaded from a URL gets the functions only after it loads. The functions can exist before your script runs, or arrive later. Handle both cases:

```
if (typeof triggerEvent === 'function') init();
else window.addEventListener('simlabready', init);
```

A call made before the functions exist throws a `ReferenceError`.

Functions

Function	Arguments	Returns / effect
`triggerEvent(id, value)`	`id`: custom event ID. `value`: optional string.	Fires the Training Builder custom event.
`playAnimation(idOrName)`	Sequence GUID or name.	Plays the sequence.
`setVariant(sceneStateId, visibilities)`	`sceneStateId`: scene state GUID. `visibilities`: optional array of `{nodeIndex, visible}`.	Applies the scene state, then sets the visibility of each listed node.
`getVariableValueByGuid(guid)`	Variable GUID.	The current value, or `undefined`.
`getVariableValueByName(name)`	Variable name.	The current value, or `undefined`.

Function	Arguments	Returns / effect
`setVariableValueByGuid(guid, value)`	Variable GUID and the new value.	Writes the variable.
`setVariableValueByName(name, value)`	Variable name and the new value.	Writes the variable.
`window.open(url)`	Absolute URL.	Opens the URL outside the panel.

triggerEvent

```
triggerEvent(id, value)
```

- `id` (string): the ID of a custom event in the Training Builder.
- `value` (string, optional): a value sent with the event.

Fires the custom event `id`. The Custom Event Triggered node in the Training Builder receives it.

```
document.getElementById('start').addEventListener('click', function () {
  triggerEvent('StartTraining', 'panel');
});
```

playAnimation

```
playAnimation(idOrName)
```

- `idOrName` (string): the GUID or the name of an animation sequence.

Plays the sequence.

```
playAnimation('Open Door');
```

setVariant

```
setVariant(sceneStateId, visibilities)
```

- `sceneStateId` (string): the GUID of a scene state.

- `visibilities` (array, optional): rows of `{nodeIndex, visible}`. `nodeIndex` is a node GUID as a string. `visible` is a boolean.

Applies the scene state. Then, for each row in `visibilities`, sets the visibility of that node.

```
setVariant('YOUR_SCENE_STATE_GUID', [
  { nodeIndex: 'YOUR_NODE_GUID', visible: false }
]);
```

getVariableValueByGuid

```
getVariableValueByGuid(guid)
```

- `guid` (string): the GUID of a variable.

Returns the current value of the variable. The type of the return value depends on the type of the variable:

- A numeric variable returns a number.
- A BOOL variable returns a boolean.
- Any other variable returns a string.
- A variable that does not exist returns `undefined`.

```
var score = getVariableValueByGuid('YOUR_VARIABLE_GUID');
if (score === undefined) console.warn('No such variable');
```

getVariableValueByName

```
getVariableValueByName(name)
```

- `name` (string): the name of a variable.

Returns the current value of the variable. The return types are the same as for `getVariableValueByGuid`.

```
var done = getVariableValueByName('StepDone');
document.getElementById('next').disabled = done !== true;
```

setVariableValueByGuid

```
setVariableValueByGuid(guid, value)
```

- `guid` (string): the GUID of a variable.
- `value`: the new value.

Writes `value` to the variable. The viewer converts `value` to the type of the variable.

```
setVariableValueByGuid('YOUR_VARIABLE_GUID', 42);
```

setVariableValueByName

```
setVariableValueByName(name, value)
```

- `name` (string): the name of a variable.
- `value`: the new value.

Writes `value` to the variable. The viewer converts `value` to the type of the variable.

```
var input = document.getElementById('speed');
setVariableValueByName('Speed', input.value);
```

window.open

```
window.open(url)
```

- `url` (string): an absolute URL.

Opens the URL outside the panel.

```
window.open('https://www.simlab-soft.com');
```

Reading variable changes

The viewer sends no event when a variable changes. To show a live value, call a getter on a timer. Panel Designer pages poll every 100 ms.

```
setInterval(function () {
  document.getElementById('speed').textContent = getVariableValueByName('Speed');
}, 100);
```

Resources and base URL

The viewer loads the HTML as a string with the base URL `http://localhost`. Relative URLs to scripts, style sheets, and images do not resolve. Use one of these instead:

- Put scripts in `<script>` elements and styles in `<style>` elements.
- Embed images as data URIs.
- Use absolute URLs.

Example page

This page uses every function. Replace the placeholders `YOUR_EVENT_ID`, `YOUR_SEQUENCE_NAME`, `YOUR_SCENE_STATE_GUID`, and `YOUR_VARIABLE_NAME` with identifiers from your scene. The commented lines also use `YOUR_NODE_GUID` and `YOUR_VARIABLE_GUID`.

```
<!DOCTYPE html>
<!--
  Starter page for a SimLab Web Browser node with Source set to HTML.
  Replace the placeholders with identifiers from your scene before you use it:
    YOUR_EVENT_ID          the ID of a custom event in the Training Builder
    YOUR_SEQUENCE_NAME     the name or GUID of an animation sequence
    YOUR_SCENE_STATE_GUID  the GUID of a scene state
    YOUR_VARIABLE_NAME     the name of a variable
  The commented lines also use YOUR_NODE_GUID and YOUR_VARIABLE_GUID.
  The viewer loads this page as a string with base http://localhost, so relative
  links to scripts, styles, and images do not resolve. Keep them inline, use data
  URIs for images, or use absolute URLs.
-->
<html>
<head>
```

```
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>SimLab panel</title>
<style>
  body {
    margin: 0;
    padding: 24px;
    font-family: "Segoe UI", Arial, sans-serif;
    font-size: 22px;
    color: #1d2330;
    background: #f4f6f9;
  }
  h1 { font-size: 30px; margin: 0 0 16px; }
  h2 { font-size: 24px; margin: 24px 0 12px; }
  .actions { display: grid; grid-template-columns: 1fr 1fr; gap: 12px; }
  button, input {
    font: inherit;
    min-height: 64px;
    border-radius: 8px;
    box-sizing: border-box;
  }
  button { border: 0; background: #2b6cd4; color: #fff; padding: 0 20px; }
  button:active { background: #1f55ab; }
  input { border: 2px solid #b8c0cc; padding: 0 14px; width: 100%; }
  form { display: grid; grid-template-columns: 1fr 1fr auto; gap: 12px; }
  .readout { font-size: 40px; font-weight: 600; }
  #status { min-height: 30px; color: #5a6475; }
</style>
</head>
<body>
  <h1>SimLab panel</h1>

  <h2>Actions</h2>
  <div class="actions">
    <button id="fireEvent" type="button">Trigger event</button>
    <button id="playSequence" type="button">Play animation</button>
    <button id="applyState" type="button">Set variant</button>
    <button id="openSite" type="button">Open website</button>
  </div>

  <h2>Live value of YOUR_VARIABLE_NAME</h2>
```

```
  <div class="readout" id="liveValue">-</div>

  <h2>Read a variable</h2>
  <form id="readForm">
    <input id="readName" placeholder="Variable name" value="YOUR_VARIABLE_NAME">
    <input id="readResult" placeholder="Value" readonly>
    <button type="submit">Read</button>
  </form>

  <h2>Write a variable</h2>
  <form id="writeForm">
    <input id="writeName" placeholder="Variable name" value="YOUR_VARIABLE_NAME">
    <input id="writeValue" placeholder="New value">
    <button type="submit">Write</button>
  </form>

  <p id="status">Waiting for the viewer...</p>

<script>
  var LIVE_VARIABLE_NAME = 'YOUR_VARIABLE_NAME';

  function showStatus(text) {
    document.getElementById('status').textContent = text;
  }

  function formatValue(value) {
    return value === undefined ? 'not found' : String(value);
  }

  function init() {
    showStatus('Connected to the viewer.');

    document.getElementById('fireEvent').addEventListener('click', function () {
      // Listen for this in the Training Builder with a Custom Event Triggered node.
      triggerEvent('YOUR_EVENT_ID', 'pressed');
      showStatus('Triggered YOUR_EVENT_ID.');
    });

    document.getElementById('playSequence').addEventListener('click', function () {
      playAnimation('YOUR_SEQUENCE_NAME');
      showStatus('Playing YOUR_SEQUENCE_NAME.');
```

```
    });

    document.getElementById('applyState').addEventListener('click', function () {
      setVariant('YOUR_SCENE_STATE_GUID');
      // Pass visibilities to show or hide nodes as well:
      // setVariant('YOUR_SCENE_STATE_GUID', [{ nodeIndex: 'YOUR_NODE_GUID', visible: false
}]);
      showStatus('Applied YOUR_SCENE_STATE_GUID.');
    });

    document.getElementById('openSite').addEventListener('click', function () {
      window.open('https://www.simlab-soft.com');
    });

    document.getElementById('readForm').addEventListener('submit', function (e) {
      e.preventDefault();
      var name = document.getElementById('readName').value;
      document.getElementById('readResult').value = formatValue(getVariableValueByName(name));
      // getVariableValueByGuid('YOUR_VARIABLE_GUID');
    });

    document.getElementById('writeForm').addEventListener('submit', function (e) {
      e.preventDefault();
      var name = document.getElementById('writeName').value;
      // The viewer converts the text to the variable's type.
      var value = document.getElementById('writeValue').value;
      setVariableValueByName(name, value);
      // setVariableValueByGuid('YOUR_VARIABLE_GUID', value);
      showStatus('Wrote ' + value + ' to ' + name + '.');
    });

    // The viewer sends no event when a variable changes, so the page polls.
    // 100 ms is the interval the Panel Designer pages use.
    var liveValue = document.getElementById('liveValue');
    setInterval(function () {
      liveValue.textContent = formatValue(getVariableValueByName(LIVE_VARIABLE_NAME));
    }, 100);
  }

  // The functions can exist before this script runs, or arrive later with the
  // simlabready event. A page loaded from a URL always gets them after it loads.
```

```
  if (typeof triggerEvent === 'function') init();
  else window.addEventListener('simlabready', init);
</script>
</body>
</html>
```