

# ISDKDOMINSTANCEBASE ADDON SDK INTERFACE

**View online:** <https://www.construct.net/en/make-games/manuals/construct-3/scripting/scripting-reference/addon-sdk-interfaces/isdkdominstancebase>

---

The `ISDKDOMInstanceBase` interface is used as a runtime base class for DOM instances (which create a HTML element) in the addon SDK. It derives from `ISDKWorldInstanceBase`.

## ISDKDOMInstanceBase APIs

---

### `_postToDOMEElement(handler, data)`

### `_postToDOMEElementAsync(handler, data)`

Post a message from the runtime instance to the DOM side. The message is received using `AddDOMElementMessageHandler()` in `DOMElementHandler`. `handler` is a string identifying the kind of message. `data` is a JSON object that is forwarded with the message to provide additional details. The async variant returns a promise that awaits an async handler on the DOM side and forwards the return value back to the runtime, which the returned promise resolves with. The non-async variant simply posts a message and ignores the result (i.e. fire-and-forget).

### `_postToDOMEElementMaybeSync(handler, data)`

As with `_postToDOMEElement()`, but when the runtime is in DOM mode, calls the DOM handler synchronously inside the call. When the runtime is in worker mode, this still posts a message which is handled later. Usually this method is not necessary, but it can be used to work around some user input restrictions in some browsers in DOM mode only.

### `_createElement(data)`

Instruct the runtime to create a DOM element for this instance. It will end up calling `CreateElement()` in `DOMElementHandler` with `data` (an optional object with additional details to create with). The runtime associates the resulting element with this instance.

### `focusElement()`

### `blurElement()`

Helper methods to manage calling `focus()` and `blur()` on the instance's associated DOM element.

### `isElementFocused()`

Helper method to identify whether the associated HTML element is currently focused.

---

### **setElementCSSStyle(prop, val)**

Helper method to set a CSS style on the instance's associated HTML element. For example

`setElementCSSStyle("font-family", "sans-serif")` will be forwarded to

`elem.style.fontFamily = "sans-serif"` on the DOM side.

---

### **setElementAttribute(attribName, value)**

### **removeElementAttribute(attribName)**

Helper method to set or remove an attribute on the instance's associated HTML element.

---

### **setElementVisible(isVisible)**

Set whether the associated HTML element is visible or invisible. This sets the CSS style

`display: none` when invisible.

---

### **\_getElementState()**

Override to return a JSON object representing the state of the DOM element, e.g. the text content. This is used by `CreateElement()` and `UpdateElementState()` to retrieve state to pass to the DOM side.

---

### **\_updateElementState()**

Send a message from the runtime to the DOM side with the element state (retrieved from `GetElementState()`). This results in a call to `UpdateState(elem, e)` on the DOM side. This is a convenient way to make sure any changes to the DOM element are applied.

---

### **\_getElementInDOMMode()**

When the runtime is in DOM mode, this returns the HTML element associated with this instance. When the runtime is in worker mode this method will throw an exception, as the DOM is not directly accessible from a Web Worker.