DOMAssistant

DOMAssistant module

Here you will find a general documentation for the DOMAssistant module.

Module documentation

The core DOMAssistant module is required and it lays the groundwork for all DOMAssistant functionality. It consists of core functionality and a few important methods to work with. These methods are:

Methods

- $
- elmsByClass
- elmsByAttribute

$()

The $ method is used to get a reference to an element. It supports either one or several string containing the id of the element(s) you want a reference too, or getting an already established element reference. If it gets an element reference, it will return that same referece, but with the difference that it has applied all extra methods to it.

Parameters

Send in one or several strings with the `id` of the element(s) you're looking, or a single element reference.

Example calls

```
// Returns element reference
$("container");
// Return an array of element references
$("container", "navigation", "content");
// Returns element reference
$(elementReference);
```

elmsByClass(className, tag)

Adds support to every element for getting elements based on their `className`. The method has a required parameter which is the desired `className`, and one optional if you want to limit the search to a certain tag. It also per default applies this method to the `document` element.

Parameters

className

Class name to search for. Required.

tag

Only search elements that have this tag name. Optional.

Example calls

```
// All calls return an array of element references
$("container").elmsByClass("mandatory");
$("container").elmsByClass("external-link",   "a");
```

elmsByAttribute(attr, attrVal, tag)

Adds support to every element for getting elements based on if they have a certain attribute. You can also specify if that attribute should have a speific value and if you want to limit the search to a certain tag. Only the first parameter specifying the attribute is required. It also

per default applies this method to the `document` element.

Parameters

attr

Attribute name to look for. Required.

attrVal

Value that the desired attribute has to have. Optional.

tag

Only search elements that have this tag name. Optional.

Example calls

```
// All calls return an array of element references
$("container").elmsByAttribute("href");
$("container").elmsByAttribute("name",   "subscription");
$("container").elmsByAttribute("type",   "text", "input");
```

DOMAssistantAJAX module

Here you will find a general documentation for the DOMAssistantAJAX module.

Module documentation

The DOMAssistantAJAX module offers basic AJAX interaction for retrieving data from a URL and then passing on the returned content to any function. The available methods is:

Methods

get

get(url, callbackFunction)

Makes a request to the specified URL and calls the set callbackFunction. The first parameter for the callbackFunction will be the responseText of the AJAX call.

Parameters

url

URL to make an AJAX call to. Required.

callbackFunction

Function name or anonymous function to call when the AJAX request is complete. Optional.

Example calls

```
// Makes an AJAX call to the my-url.com and then calls callbackFunctionName
DOMAssistant.AJAX.get("my-url.com", callbackFunctionName);

// Makes an AJAX call to the my-url.com and then calls an anonymous function
DOMAssistant.AJAX.get("index.htm", function(returnValue){
        alert(returnValue);
});
```

DOMAssistantContent module

Here you will find a general documentation for the DOMAssistantContent module.

Module documentation

The DOMAssistantContent module offers various methods for adding and removing content and elements to the page. These methods are:

Methods

- prev
- next
- create
- setattributes
- addcontent
- replacecontent
- remove

prev()

Gets a reference to the previous HTML element, automatically bypassing any text nodes that might in between.

Parameters
None.

Example calls

```
// Returns element reference
$("container").prev();
```

next()

Gets a reference to the next HTML element, automatically bypassing any text nodes that might in between.

Parameters
None.

Example calls

```
// Returns element reference
$("container").next();
```

create(name, attr, append, content)

Creates an element, and optionally sets attributes on it, appends it to the current element

and adds content to it.

Parameters

name

Tag name for the new element. Required.

attr

An object containing attributes and their values. Optional.

append

Boolean if the new element should be appended right away. Optional.

content

A string or HTML object that will become the content of the newly created element. Optional.

Example calls

```
// Creates a DIV element
$("container").create("div");
// Creates a DIV element and sets some attributes
$("container").create("div",  {
        id : "my-div",
        className : "my-class"
});
// Creates a DIV element and sets its content to the text "Hello!"
$("container").create("div",  null, false, "Hello!");
/*
        Creates a DIV element, sets its content to the text
        "Hi there!" and appends it to the parent element
*/
$("container").create("div",  {
        id : "my-div",
        className : "my-class"
}, true, "Hi there!");
```

setAttributes(attr)

Sets attributes on the current element.

Parameters

attr

An object containing attributes and their values. Required.

Example calls

```
// Sets some attributes on the element named "container"
$("container").setAttributes({
        id : "my-div",
        className : "my-class"
});
```

addContent(content)

Adds content to the current element.

Parameters

content

Can either be a string, which will then be applied using `innerHTML`, or an HTML object that will be applied using `appendChild`.

Example calls

```
// Adds content, using innerHTML, to the element named "container"
$("container").addContent("<p>A  new paragraph</p>");
// Adds content, using an element, to the element named "container"
$("container").addContent(document.createElement("p"));
```

replaceContent(newContent)

Replaces the content of the current element with new content.

Parameters

newContent

Can either be a string, which will then be applied using `innerHTML`, or a HTML object that will be applied using `appendChild`.

Example calls

```
// Replaces content in the element named "container" (innerHTML)
$("container").replaceContent("<p>A  new paragraph</p>");
// Replaces content in the element named "container" (new element)
$("container").replaceContent(document.createElement("p"));
```

remove()

Removes the current element.

Parameters
None.

Example calls

```
// Removes the element named "container"
$("container").remove();
```

DOMAssistantCSS module

Here you will find a general documentation for the DOMAssistantCSS module.

Module documentation

The DOMAssistantCSS module offers various methods for adding and removing CSS classes, checking if an element has a certain class and getting the rendered style for a specific property on an element. These methods are:

Methods

addClass
removeClass
hasClass
getStyle

addClass(className)

Adds a class name to the current element, unless it already exists.

Parameters

className

Desired class name to be added. Required.

Example calls

```
// Adds a class name to the element named "container"
$("container").addClass("selected");
```

removeClass(className)

Removes a class name from the current element.

Parameters

className

Desired class name to remove. Required.

Example calls

```
// Removes a class name from the element named "container"
$("container").removeClass("selected");
```

hasClass(className)

Checks whether the current element has a certain class name.

Parameters

className

Class name to check if it exists on the current element. Required.

Example calls

```
/*
        Checks whether a class name of "selected"
        exists or not on the current element
*/
$("container").hasClass("selected");
```

getStyle(cssRule)

Gets the rendered style for a certain CSS property on the current element, no matter if it was applied inline or from an external stylesheet. Note: make sure to look for the specific property instead of general ones. I.e. `background-color` instead of `backgrund` etc.

Parameters

cssRule

CSS property to check value of. Use CSS syntax for the property, i.e. `background-color` instead of `backgroundColor`. Required.

Example calls

```
/*
        Checks what value the rendered CSS property
        background-color is on the current element
*/
$("container").getStyle("background-color");
```

DOMAssistantEvents module

Here you will find a general documentation for the DOMAssistantEvents module.

Module documentation

The DOMAssistantEvents module offers various methods for adding and removing handlers for one or several events on an element, It also contains functionality for stopping default actions and bubbling of events. The methods are:

Methods

addEvent
removeEvent
preventDefault
cancelBubble

addEvent(evt, func)

Adds an event handler to the current element. Multiple event handlers are supported, and the receiving funtion will have an event object reference and a `this` reference to the element it occurred on, no matter what web browser. For accessibility reasons, please make sure to only apply click events to elements that can handle them without JavaScript enabled.

Parameters

evt

Event to apply, specified as a string, without the "on" prefix.

func

Function to handle the event, specified as a function reference (without parentheses) or an anonymous function.

Example calls

```
// Adds a click event to the element named "get-listing"
$("container").addEvent("click", getListing);
// Adds a click event to the element named "get-listing"
$("container").addEvent("click", function (){
        alert("Hello darling!");
});
```

removeEvent(evt, func)

Removes an event handler from the current element. Works only for function references, and not anonymous functions.

Parameters

evt

Event to remove, specified as a string, without the "on" prefix.

func

Function to stop from handling the event, specified as a function reference (without parentheses).

Example calls

```
// Removes a click event from the element named "get-listing"
$("container").removeEvent("click",  getListing);
```

preventDefault(evt)

Prevents the default action of an event. Can be called from any function, and is not a method of any element.

Parameters

evt

Event to prevent the default action of.

Example calls

```
// Prevents the default behavior of an event
DOMAssistant.preventDefault(eventReference);
```

cancelBubble(evt)

Cancels the bubbling of an event. Can be called from any function, and is not a method of any element.

Parameters

evt

Event to cancel the bubbling of.

Example calls

```
// Cancels the bubbling of an event
DOMAssistant.cancelBubble(eventReference);
```

DOMAssistantLoad module

Here you will find a general documentation for the DOMAssistantLoad module.

Module documentation

The DOMAssistantLoad module offers a way to call a number of functions as soon as the DOM has loaded, as opposed to waiting for all images and other external files to completely load. It was inspired and influenced by Dean Edwards, Matthias Miller, and John Resig: window.onload (again).

Methods

DOMReady

DOMReady()

From any file, just call the DOMReady method with desired functions and they will be executed as soon as the DOM has loaded.

Parameters
Send in any number of function references, anonymous functions or strings with function names and parentheses.

Example calls

```
// Calls myFunc when the DOM is loaded
DOMAssistant.DOMReady(myFunc);

// Calls myFunc with one parameter when the DOM is loaded
DOMAssistant.DOMReady("myFunc('Some text')");

// Calls myFunc and anotherFunction when the DOM is loaded
DOMAssistant.DOMReady(myFunc, "anotherFunction()");

// Calls myFunc and an anonymous function when the DOM is loaded
DOMAssistant.DOMReady(myFunc, function(){
        // Perform some magic
});
```