



Vulcan.js v2.8 - Documentation

Author: George Delaportas

License: Open Software License (OSL 3.0)

Purpose: Utility and support library for general JavaScript operations in the **micro-MVC** ecosystem



Overview

vulcan.js is a lightweight, modular JavaScript utility framework designed to streamline validation, event control, DOM interaction, graphics management, and system resource loading. It acts as a foundation for JavaScript logic within the **micro-MVC** structure.

Core Modules

1. **validation**

Contains four submodules for different validation types.

♦ **alpha**

- `is_string(val)`
- `is_symbol(val)`
- `is_blank(val)`

♦ **numerics**

- `is_number(val)`
- `is_integer(val)`
- `is_float(val)`

♦ **misc**

- `is_undefined(val)`
- `is_nothing(val)` (null or empty string)
- `is_bool(val)`
- `is_array(val)`
- `is_function(val)`
- `is_object(val)`
- `is_invalid(val)` (undefined or null or "")

♦ **utilities**

- `is_email(val)`
- `is_phone(val)`
- `reg_exp(pattern, expression)`

2. **events**

Handles event binding and unbinding in a controlled, trackable way using an internal registry.

✖ **Internal Structure**

- **caller_model**: tracks caller ID and associated objects
- **object_events_model**: binds objects and their events
- **insert()**: internal, used for controlled binding
- **fetch()**: retrieves handlers for removal

♦ **Public Methods**

- `attach(caller_id, object, func, handler)`
Adds an event listener with traceability.
- `detach(caller_id, object, func, handler)`
Removes a specific event handler using metadata.

🛑 **Caveat:** Requires objects to be native DOM elements.

3. conversions

♦ `object_to_array(conversion_mode, model)`

- If `conversion_mode == true`, returns `[[key, value], ...]`
- Else, returns `[value1, value2, ...]`

♦ `replace_link(mode, text, attributes, url_info)`

- `mode = 1`: auto-link detection via regex (plain text to `<a>`)
- `mode = 2`: use Twitter-like `url_info` objects for smart linking
- `attributes`: optional map (e.g., `{target: '_blank'}`)

4. graphics

♦ `pixels_value(pixels)`

Converts a string like `'120px'` → `120` (integer)

♦ `apply_theme(directory, theme, fail_on_existing = true, clear_cache = true)`

- Dynamically loads a CSS file from the specified directory
- Prevents double-loading and allows optional cache busting

♦ `clear_theme(theme)`

Removes a specific theme (CSS link) from DOM.

5. misc

♦ `active_language()`

Returns the first two characters of the URL pathname.

♦ `contains(subject, list)`

Returns true if the `subject` is in `list`.

♦ `sort(array, mode, by_property)`

- `mode`: 'asc' | 'desc'
- `by_property`: optional object property to sort by.

⚠ Fails if array or `by_property` is invalid.

6. objects

DOM querying methods.

- `by_tag(tag)` → `NodeList`
- `by_id(id)` → `Element`
- `by_class(class)` → `NodeList`

♦ `selectors`

- `selectors.first(query)` → `document.querySelector(query)`
- `selectors.all(query)` → `document.querySelectorAll(query)`

7. system

Manages JavaScript and CSS files at runtime.

- ♦ `require(path, name, clear_cache = true)`

Dynamically appends `<script>` with cache-busting optional.

- ♦ `source_exists(name, tag_type, attr)`

Checks if `<script>` or `<link>` exists by file name substring match.

- ♦ `remove_source(name, tag_type, attr)`

Removes matching DOM resource tag (e.g., `<script>`, `<link>`)



Usage Examples

```
var utils = new vulcan();
```

```
// Validation
```

```
utils.validation.numerics.is_integer(42); // true
```

```
utils.validation.utilities.is_email("hello@world.com"); // true
```

```
// Event attach/detach
```

```
utils.events.attach("userForm", someElement, "click", handlerFn);
```

```
utils.events.detach("userForm", someElement, "click", handlerFn);
```

```
// Dynamic theme
```

```
utils.graphics.apply_theme("/themes", "dark");
```

```
// Dynamic script loader
```

```
utils.system.require("/scripts", "analytics", true);
```

```
// Convert object to array
```

```
utils.conversions.object_to_array(true, { a: 1, b: 2 });
```

```
// returns [['a', 1], ['b', 2]]
```