Framework Functions

Documentation Migration
Beginning in version 1.3 of Infusion, these Framework Functions will be documented in our new Documentation space. This migration is currently in progress, and will be complete in time for the 1.3 release. At that time, this page will be deprecated in favor of the new pages. If you have any comments on the new documentation format, feel free to let us know by emailing the infusion-users mailing list.

The Fluid framework includes a number of functions and utilities that component developers can avail themselves of. These are described below.

fluid.js

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.accumulate`

fluid.accumulate(list, fn, arg)

Description

Scans through a list of objects, "accumulating" a value over them (may be a straightforward "sum" or some other chained computation). The results will be added to the initial value passed in by "arg" and returned.

To a Google developer, this would be "reduce", from the "map/reduce" pairing.

Arguments

- list {Array}: The list of objects to be accumulated over
- fn {Function}: An "accumulation function" accepting the signature (object, total, index) where object is the list member, total is the "running total" object (which is the return value form the previous function), and index is the index number.
- arg {Object}: The initial value for the "running total" object

Return Value

Object: the modified "arg" object

New in v1.3: fluid.allocateGuid()

Please see fluid.allocateGuid in the new Documentation space.

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.allocateSimpleId`

fluid.allocateSimpleId(element)

Description

Allocate an id to the supplied element if it has none already, by a simple scheme resulting in ids "fluid-id-nnnn" where nnnn is an increasing integer.

Arguments

- element {Element}: The element to place the id on.

Return Value

String: representing the id

New in v1.3: fluid.arrayToHash()

Please see fluid.arrayToHash in the new Documentation space.

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.byId`

fluid.byId(id, dokkument)

Description

Quickly retrieves an element, given its id.

Arguments

- id {Object}: The id of the DOM node to find
- dokkument {Document}: the document in which it is to be found (if left empty, use the current document)

Return Value

Element: if it exists or
null: if there are no elements with the given id

Reference

See also: fluid:fluid.jById

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.clear`

fluid.clear(target)

Description

Clears an object or array of its contents. For objects, each property is deleted.

Arguments

- target {Object|Array}: Either an object or an array whose contents you wish to clear

Return Value

No Return Value

fluid.COMPONENT_OPTIONS

Description

A special "marker object" which is recognized as one of the arguments to fluid:fluid.initSubComponents. This object is recognized by reference equality - where it is found, it is replaced in the actual argument position supplied to the specific subcomponent instance, with the particular options block for that instance attached to the overall "that" object.

In essence, it tells fluid.initSubComponents to look at the parent component's options block. If options were specified for this subComponent, they will be merged in for use by that subcomponent.

Constant Value

Constant: {}

Reference

See also: fluid:fluid.initSubComponents

New in v1.3: fluid.computeNickName()

Please see fluid.computeNickName in the new Documentation space.

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.container`

fluid.container(containerSpec, fallible)

Description

Fetches a single container element and returns it as a jQuery.

Arguments

- containerSpec {String|jQuery|Element}: An id string, a single-element jQuery, or a DOM element specifying a unique container.
- New in v1.3: fallible {Boolean}: `true` if an empty container is to be reported as a valid condition.

Return Value

jQuery: a single-element jQuery of container

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.copy`

fluid.copy(tocopy)

Description

Performs a deep copy (clone) of the object passed in the argument

Arguments

- tocopy {Object}: the object to be copied

Return Value

Object: the copied object

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.createDomBinder`

fluid.createDomBinder(container, selectors)

Description

Creates a new DOM Binder instance, used to locate elements in the DOM by name.

Arguments

- container {Object}: The root element in which to locate named elements
- selectors {Object}: A collection of named jQuery selectors

Return Value

Object: representing the DOM Binder and containing all of its functions (i.e. locate, fastLocate, clear, and refresh)

Reference

See: DOM Binder

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.defaults`

fluid.defaults()

Description

Centrally stores and retrieves a component's default settings.

Arguments

- {Boolean}: (optional) if true, manipulate a global option (for the head component) rather than instance options
- {String}: componentName the name of the component
- {Object}: (optional) a container of key/value pairs to set

Return Value

Object: If the object is passed in the argument, this is added to the central store and then returned. If not, the current object in the store is returned.

Reference

See: Markup Configuration and Selectors, Fluid Component API
See also: fluid:fluid.mergeComponentOptions

New in v1.3: fluid.demands()

Please see fluid.demands in the new Documentation space.

New in v1.3: fluid.each()

Please see fluid.each in the new Documentation space.

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.emptySubcomponent`

fluid.emptySubcomponent(options)

Description

Construct a dummy or "placeholder" subcomponent, that optionally provides empty implementations for a set of methods.

Arguments

- options {Anything}: can be anything, will use jQuery.makeArray to turn it into an array

Return Value

Object: the empty subcomponent

fluid.event.getEventFirer(unicast, preventable)

Description

Constructs an "event firer" object which can be used to register and unregister listeners, to which "events" can be fired. These events consist of an arbitrary function signature.

Arguments

- unicast {Boolean}: If true, this is a "unicast" event which may only accept a single listener.
- preventable {Boolean}: If true, the return value of each handler will be checked for false in which case further listeners will be short circuited, and this will be the return value of fire().

Return Value

Object:

```
{
    addListener: function(listener, namespace, exclusions),
    removeListener: function(listener),
    fire: function()
}
```

Reference

See: Infusion Event System

New in v1.3: fluid.event.identifyListener()

Please see fluid.event.identifyListener in the new Documentation space.

New in v1.3: fluid.expandOptions()

Please see fluid.expandOptions in the new Documentation space.

New in v1.3: fluid.expect()

Please see fluid.expect in the new Documentation space.

fluid.expectFilledSelector(result, message)

Description

Expect that an output from the DOM binder has resulted in a non-empty set of results. If none are found this function will fail with a diagnostic message, with the supplied message prepended.

Arguments

- result {jQuery}: The return value from the DOM Binder
- message {String}: message to report if the results are empty

Return Value

No Return Value

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.fail`

fluid.fail(message)

Description

Causes an error message to be logged to the console and a real runtime error to be thrown.

Arguments

- message {String|Error}: The error message to log

Return Value

No Return Value

fluid.fetchResources()

Please see fluid.fetchResources in the new Documentation space.

New in v1.3: fluid.fetchResources.primeCacheFromResources()

Please see fluid.fetchResources.primeCacheFromResources in the new Documentation space.

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.find`

fluid.find(list, fn, deflt)

Description

Scans through a list of objects, terminating on and returning the first member which matches a predicate function.

Arguments

- list In v1.2: {Array}: The list of objects to be searched.
  In v1.3: {Arrayable or Object}: The list or hash of objects to be searched.
- fn {Function}: A predicate function, acting on a list member. A predicate which returns any value which is not null or undefined will terminate the search. The function has the signature (object, index).
- deflt {Object}: A value to be returned in the case no predicate function matches a list member. The default will be the natural value of undefined.

Return Value

Object: the first object in the list that matches the predicate function, or deflt if nothing does

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.findAncestor`

fluid.findAncestor(element, test)

Description

Finds the nearest ancestor of the element that passes the test.

Arguments

- element {Element}: DOM element
- test {Function}: A function which takes an element as a parameter and returns true or false for some test

Return Value

Element: the nearest ancestor that passes the test

~~fluid.findKeyInObject~~

Description

Deprecated

Reference

See: fluid:fluid.keyForValue

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.freshContainer`

fluid.freshContainer(tocopy)

Description

Return an empty container as the same type as the argument (either an array or hash).

Arguments

- tocopy {Object}: The container to copy

NEW DOCUMENTATION: Please visit our new documentation space for updated documentation on `fluid.formatMessage`

fluid.formatMessage(messageString, args)

Description

Expand a message string with respect to a set of arguments, following a basic subset of the Java Message Format rules.

The message string is expected to contain replacement specifications such as {0}, {1}, {2}, etc.

Arguments

- messageString {String}: The message key to be expanded
- args {String|Array of Strings}: An array of arguments to be substituted into the message.

Return Value

Return: String (the formatted String)

Reference

See: http://java.sun.com/j2se/1.4.2/docs/api/java/text/MessageFormat.html

new in v1.3: fluid.get(root, EL)

Description

Evaluates an EL expression by fetching a dot-separated list of members recursively from a provided root.

Arguments

- root {Object}: The root data structure in which the EL expression is to be evaluated
- EL {String}: The EL expression to be evaluated

Return Value

The fetched data value

Reference

See also: #fluid.set, fluid:fluid.model.getGlobalValue

fluid.getGlobalValue(path, env)

Description

Evaluates an EL expression by fetching a dot-separated list of members recursively from the global environment.

Arguments

- path {String}: The EL expression to be evaluated
- env {Object}: An optional "environment" if it contains any members at top level, will take priority over the global environment.

Return Value

The fetched data value.

Reference

See also: fluid:fluid.get

fluid.getId(element)

Description

Retrieves the id attribute from a jQuery or pure DOM element

Arguments

- element {jQuery|Element}: The element to return the id attribute for

Return Value

String: representing the id

fluid.identity(element), fluid.dumpEl(element), fluid.renderTimestamp (element)

Description

A basic utility that returns its argument unchanged.

Arguments

- element {jQueryable}: can be an element, selector, function that returns an element, or a jQuery.

Return Value

The original element.

New in v1.3: fluid.initDependent()

Please see fluid.initDependent in the new Documentation space.

New in v1.3: fluid.initDependents()

Please see fluid.initDependents in the new Documentation space.

fluid.initDomBinder(that)

Description

Creates a new DOM Binder instance for the specified component and mixes it in.

Arguments

- that {Object}: The component instance to attach the new DOM Binder to

Return Value

No Return Value

Reference

See: DOM Binder

fluid.initLittleComponent(name, options)

Description

Creates a new "little component": a that-ist object with options merged into it by the framework. This method is a convenience for creating small objects that have options but don't require full View-like features such as the DOM Binder or events

Arguments

- name {String}: The name of the little component to create
- options {Object}: user-supplied options to merge with the defaults

fluid.initSubcomponent(that, className, args)

Description

Provides a generic way of instantiating and configuring any "subcomponents" that may be a part of, or used by, a parent component. For example, this function is used by the Inline Edit component to instantiate the Undo decorator, a subcomponent that provides Undo functionality to components that support it.

Arguments

- that {Component}: The top-level component for which subcomponents are to be instantiated. It contains specifications for these subcomponents in its options structure.
- className {String}: The "class name" or "category" for the subcomponents to be instantiated. A class name specifies an overall "function" for a class of subcomponents and represents a category which accept the name signature of instantiation arguments.
- args {Array of Object}: The instantiation arguments to be passed to each constructed subcomponent. These will typically be members derived from the top-level "that" or perhaps globally discovered from elsewhere. One of these arguments may be fluid.COMPONENT_OPTIONS in which case this placeholder argument will be replaced by instance specific options configured into the member of the top-level options structure named for the className.

Return Value

Component: a single component

Reference

See: Subcomponents, fluid:fluid.initSubcomponents

fluid.initSubcomponents(that, className, args)

Description

Initialize all the "subcomponents" which are configured to be attached to the supplied top-level component, which share a particular "class name".

Arguments

- that {Component}: The top-level component for which sub-components are to be instantiated. It contains specifications for these subcomponents in its options structure.
- className {String}: The "class name" or "category" for the sub-components to be instantiated. A class name specifies an overall "function" for a class of sub-components and represents a category which accept the name signature of instantiation arguments.
- args {Array of Object}: The instantiation arguments to be passed to each constructed sub-component. These will typically be members derived from the top-level "that" or perhaps globally discovered from elsewhere. One of theses arguments may be fluid.COMPONENT_OPTIONS in which case this placeholder argument will be replaced by instance specific options configured into the member of the top-level options structure named for the className.

Return Value

Array: of sub-components

Reference

See also: fluid:fluid.COMPONENT_OPTIONS, fluid:fluid.initSubcomponent

fluid.initView(componentName, container, userOptions)

Description

The central initialization method called as the first act of every Fluid component. This function automatically merges user options with defaults, attaches a DOM Binder to the instance, and configures events.

Arguments

- componentName {String}: The unique "name" of the component, which will be used to fetch the default options from store. By recommendation, this should be the global name of the component's creator function.
- container {jQueryable}: A specifier for the single root "container node" in the DOM which will house all the markup for this component. Can be an element, selector, function that returns an element, or a jQuery.
- userOptions {Object}: The configuration options for this component.

Return Value

Component: the initialized component

fluid.instantiateFirers(that, options)

Description

Sets up a component's declared events.

Events are specified in the options object by name. There are three different types of events tat can be specified.

1. An ordinary multicast event, specified by "null"
2. A unicast event, which allows only one listener to be registered
3. A preventable event.

Arguments

- that {Object}: The component
- options {Object}: The component's options structure, containing the declared event names and types.

Return Value

No Return Value

Reference

See: Infusion Event System

New in v1.3: fluid.invoke()

Please see fluid.fetchResources.invoke in the new Documentation space.

fluid.invokeGlobalFunction(functionPath, args, environment)

Description

Allows for the calling of a global function from an EL expression "functionPath", with the arguments "args". For advanced use cases, the final argument "environment" can be used to specify a set of overrides lookups for certain root path segments.

Arguments

- functionPath {String}: An EL expression, referred to the global object root.
- args {Array}: An array of arguments to be applied to the function, specified in functionPath
- environment {Object}: (optional) A hash of root path segments to objects that can be used to override lookups to the global object. This would typically be used by the Fluid framework itself to direct lookups to the correct version of the root `fluid` object.

Return Value

Function: the function represented by "functionPath" with the "args" applied

Reference

See: fluid:fluid.initSubcomponents
See also: fluid:fluid.get

New in v1.3: fluid.iota()

Please see fluid.iota in the new Documentation space.

fluid.isArrayable (totest)

Description
Determines whether the supplied object can be treated as an array, by iterating an index towards its length. The test functions by detecting
a property named "length" which is of type "number", but excluding objects
which are themselves of type "string".

Arguments

- totest {object}: The object to be tested.

Return Value

- {Boolean} true if value can be treated as an array, otherwise false.

Applicable Infusion Versions

- v1.3

New in v1.3: fluid.isMarker()

Please see fluid.isMarker in the new Documentation space.

fluid.isPrimitive(value)

Description

Determines if the passed value is of a primitive type

Arguments

- value {Object}: The value to be tested

Return Value

Boolean: true if value is a primitive, otherwise false

fluid.jById(id, dokkument)

Description

Will create a jQuery object from the id of a DOM node.

Arguments

- id {String}: id of the DOM node to find
- dokkument {}: the document in which it is to be found (if left empty, use the current document)

Return Value

Object: a jQuery Object for the id or an empty list of the id doesn't exist

Reference

See also: fluid:fluid.byId

fluid.keyForValue(obj, value)

Description

Searches through the supplied object for the first value which matches the one supplied.

Arguments

- obj {Object}: The Object to be searched through
- value {Object}: The value to be found. This will be compared against the object's member using === equality

Return Value

String: representing the key for the passed value
null: if no key is found

Reference

See also: fluid:~~fluid:fluid.findKeyInObject~~

fluid.log(str)

Description

Log a message to a suitable environmental console. If the standard "console" stream is available, the message will be sent there - otherwise either the YAHOO logger or the Opera "postError" stream will be used. Logging must first be enabled with a call to...

```
fluid.setLogging(true)
```

Arguments

- str {String}: the message to log

Return Value

No Return Value

Reference

See also: fluid:fluid.setLogging

fluid.merge(policy, target)

Description

Merge a collection of options structures onto a target, following an optional policy. This function is typically called automatically, as a result of an invocation of `fluid.iniView()`. The behaviour of this function is explained more fully on the page Options Merging for Infusion Components.

Arguments

- policy {Object/String}: A "policy object" specifying the type of merge to be performed. If policy is of type {String} it should take on the value "reverse" or "replace" representing a static policy. If it is an Object, it should contain a mapping of EL paths onto these String values, representing a fine-grained policy. If it is an Object, the values may also themselves be EL paths representing that a default value is to be taken from that path.
  New in v1.3: `fluid.merge()` now supports a policy of "preserve," which will preserve the actual source object. Any defaults will be merged into this object.
- target {Object}: The options structure which is to be modified by receiving the merge results.
- options1, options2, ... {Object} an arbitrary list of options structure which are to be merged "on top of" the `target`. These will not be modified.

Return Value

The modified target.

Reference

See: Options Merging for Infusion Components
See also: fluid:fluid.initView
See also: fluid:fluid.defaults

fluid.mergeComponentOptions(that, componentName, userOptions)

Description

Merges the component's declared defaults, as obtained from fluid:fluid.defaults, with the user's specified overrides.

New in v1.3 (Sneak Peek): Will invoke any expanders that may be specified.

Arguments

- that {Object}: The instance to attach the options to
- componentName {String}: The unique "name" of the component, which will be used to fetch the default options from store. By recommendation, this should be the global name of the component's creator function.
- userOptions {Object}: The user-specified configuration options for this component

Return Value

No Return Value

Reference

See also: fluid:fluid.defaults
New in v1.3 (Sneak Peek) See also: fluid.expandOptions

fluid.mergeListeners(events, listeners)

Description

Attaches the user's listeners to a set of events.

Arguments

- events {Object}: A collection of named event firers
- listeners {Object}: Optional listeners to add

Return Value

No Return Value

fluid.messageLocator(messageBase)

(moved to `fluidRenderer.js`)

Description

Converts a data structure consisting of a mapping of keys to message strings, into a "messageLocator" function which maps an array of message codes, to be tried in sequence until a key is found, and an array of substitution arguments into a substituted message string.

Arguments

- messageBase {Object}: The data structure of keys to message strings

Return Value

String: the substituted message string or an error message indicating that the message string for the key wasn't found

New in v1.3: fluid.messageResolver()

Please see fluid.messageResolver in the new Documentation space.

fluid.model.composePath(prefix, suffix)

Description

Assembles a path by concatenating two String values, representing portions of the path, with a (.) separating them.

Arguments

- prefix {String}: A string representing the beginning of the path
- suffix {String}: A string representing the end of the path

Return Value

String: representing the composed path

New in v1.3: fluid.model.composeSegments()

Please see fluid.model.composeSegments in the new Documentation space.

fluid.model.copyModel(target, source)

Description

Copy a source "model" onto a target

Arguments

- target {Object}: the object to be extended with the source
- source {Object}: the object to add onto the target

Return Value

No Return Value

~~fluid.model.getBeanValue(root, EL)~~

Deprecated: Please see #fluid.get instead.

Description

Evaluates an EL expression by fetching a dot-separated list of members recursively from a provided root.

Arguments

- root {Object}: The root data structure in which the EL expression is to be evaluated
- EL {String}: The EL expression to be evaluated
- v1.2 and earlier: environment {Object}: (optional) if it contains any members at top level, will take priority over the root data structure. Will change the scope of the evaluation.
- New in v1.3: config {Array of function}: (optional) An array of functions that will be used to filter or modify the retrieved value before it is returned.

Return Value

The fetched data value

Reference

See also: fluid:fluid.model.setBeanValue

fluid.model.getPenultimate(root, EL, environment, create)

Description

Arguments

- root {Object}: The root data structure in which the EL expression is to be evaluated
- EL {String}: The EL expression to be evaluated
- environment {Object}: (optional) if it contains any members at top level, will take priority over the root data structure. Will change the scope of the evaluation.
- create {boolean}: A boolean indicating whether or not the element hierarchy should be created if necessary for the given EL path.

Return Value

An object containing the root and the fetched data value.

Reference

See also: fluid:fluid.get

fluid.model.parseEL(EL)

Description

Parse an EL expression separated by periods (.) into its component segments.

Arguments

- EL {String}: The EL expression to be split

Return Value

Array: of Strings representing the path expressions

~~fluid.model.setBeanValue(root, EL, newValue)~~

Deprecated: Please see #fluid.get instead

Description

Evaluates an EL expression by fetching a dot-separated list of members recursively from a provided root, and sets the value of the evaluated field.

Arguments

- root {Object}: The root data structure in which the EL expression is to be evaluated
- EL {String}: The EL expression to be evaluated
- newValue {Object}: the value to set the evaluated field to
- v1.2 and earlier: environment {Object}: (optional) if it contains any members at top level, will take priority over the root data structure. Will change the scope of the evaluation.
- New in v1.3: config {Array of function}: (optional) An array of functions that will be used to filter or modify the value before it is saved.

Return Value

No Return Value

Reference

See also: fluid:fluid.model.getBeanValue

New in v1.3: fluid.model.resolvePathSegment()

Please see fluid.model.resolvePathSegment in the new Documentation space.

New in v1.3: fluid.model.transformWithRules()

Please see fluid.model.transformWithRules in the new Documentation space.

New in v1.3: fluid.prettyPrintJSON()

Please see fluid.prettyPrintJSON in the new Documentation space.

New in v1.3: fluid.progressiveChecker()

Please see fluid.progressiveChecker in the new Documentation space.

fluid.registerGlobal(functionPath, func, env), fluid.registerGlobalFunction (functionPath, func, env)

Description

Registers a new global function at a given path (currently assumes that it lies within the fluid namespace).

Arguments

- functionPath {String}: The desired EL patch.
- func {Function}: The function to register
- env {Object}

Return Value

Object: the modified "list". *note that the original list is returned and it, itself is modified*

fluid.registerNamespace(naimspace, env)

Description

Ensures that an entry in the global namespace exists. If it is not found, it will be created.

Arguments

- naimspace {String}: The namespace.
- env {Object}

Return Value

The found or created object.

NEW for v1.3: Please visit our new documentation space for updated documentation on fluid.remove_if

v1.2.x and earlier:

fluid.remove_if(list, fn)

Description

Can traverse through a list of objects, removing those which match a predicate. Similar to jQuery.grep, only acts on the list in-place by removal, rather than by creating a new list by inclusion.

Arguments

- list {Array}: The list of objects to be scanned over
- fn {Function}: A predicate function determining whether an element should be removed. This accepts the standard signature (object, index) and returns a "truthy" result in order to determine that the supplied object should be removed from the list.

Return Value

Object: the modified "list". *note that the original list is returned and it, itself is modified*

New in v1.3: fluid.render()

Please see fluid.render in the new Documentation space.

New in v1.3: fluid.renderer.createRendererFunction()

Please see fluid.renderer.createRendererFunction in the new Documentation space.

New in v1.3: fluid.renderer.makeProtoExpander()

Please see fluid.renderer.makeProtoExpander in the new Documentation space.

New in v1.3: fluid.renderer.mergeComponents()

Please see fluid.renderer.mergeComponents in the new Documentation space.

New in v1.3: fluid.renderer.selectorsToCutpoints()

Please see fluid.renderer.selectorsToCutpoints in the new Documentation space.

new in v1.3: fluid.set(root, EL, newValue)

Description

Sets the value at an EL expression within a model.

Arguments

- root {Object}: The root data structure in which the EL expression is to be set
- EL {String}: The EL expression to be set
- newValue {String}: The new value

Return Value

none

Reference

See also: fluid:fluid.get

fluid.setLogging(enabled)

Description

Method to allow the user to enable logging (off by default)

Arguments

- enabled {Boolean}: True to enable logging, False to disable logging

Return Value

No Return Value

Reference

See also: fluid:fluid.log

fluid.stringTemplate(template, values)

Please see fluid.stringTemplate in the new Documentation space

fluid.transform(list)

Description

Transforms a list of objects, by one or more functions. Similar to jQuery.map, it will only accept an arbitrary list of transformation functions.

Arguments

- list {Array}: The initial array of objects to be transformed.
- functions {Function}: An arbitrary number of optional further arguments, all of type Function, accepting the signature (object, index), where object is the "list" member to be transformed, and index is its index in "list". Every function passed in the arguments, will be applied onto every object in "list". After a function is called on an object in "list", that object is replaced by the return value of the function.

Return Value

Array: the transformed set of objects

fluid.unwrap(obj)

Description

If the object is a jQuery, then return the first DOM element within it.

Arguments

- obj {jQuery}: The jQuery instance to unwrap into a pure DOM element

Return Value

Element: if obj is a jQuery containing a single element
Object: if obj is not a jQuery containing a single element, it will return obj

Reference

See also: fluid:fluid.wrap

fluid.version

Description

A String constant representing the version of the Fluid Framework in use.

Constant Value

Constant: String (i.e. "Infusion 1.0")

fluid.wrap(obj)

Description

Wraps an object in a jQuery if it isn't already one. This function is useful since it ensures to wrap a null or otherwise falsy argument to itself, rather than the jQuery default of returning the overall document node.

Arguments

- obj {Object}: the object to wrap inside of a jQuery

Return Value

jQuery: obj wrapped inside a jQuery object
Object: if obj is falsy, obj itself is returned.

Reference

See also: fluid:fluid.unwrap

FluidDOMUtilities.js

fluid.dom.cleanseScripts(element) As of v1.2, moved to ReordererDomUtilities.js

Description

Cleanses the children of a DOM node by removing all <script> tags. This is necessary to prevent the possibility that these blocks are reevaluated if the node were reattached to the document.

Arguments

- element {Element}: The parent element to begin removing <script> tags from.

Return Value

No Return Value

fluid.dom.cleanseScripts.MARKER As of v1.2, moved to ReordererDomUtilities.js

Description

Used to indicate that the DOM node has been stripped of all <script> tags

Constant Value

Constant: "fluid-scripts-cleansed"

Reference

See also: fluid:fluid.dom.cleanseScripts

fluid.dom.computeAbsolutePosition(element) As of v1.2, moved to ReordererDomUtilities.js

Description

Returns the absolute position of a supplied DOM node in pixels. Implementation taken from quirksmode http://www.quirksmode.org/js/findpos.html

Arguments

- element {Element}: The element whose location is returned

Return Value

Array: containing the absolute position coordinates

```
[curleft, curtop]
```

Reference

See: http://www.quirksmode.org/js/findpos.html

fluid.dom.getElementText(element)

Description

Gets the element text from the supplied DOM node

Arguments

- element {Element}: the element to return the text from

Return Value

String: the element text

fluid.dom.insertAfter(newChild, refChild) As of v1.2, moved to ReordererDomUtilities.js

Description

Inserts newChild as the next sibling of refChild.

Arguments

- newChild {Element}: the new element to insert
- refChild {Element}: the element to insert newChild after

Return Value

No Return Value

fluid.dom.isContainer(container, containee)

Description

Checks if the specified container is actually the parent of containee.

Arguments

- container {Element}: the potential parent
- conatainee {Element}: the child in question

Return Value

Boolean: true if container is a parent of containee, otherwise false.

fluid.dom.isIgnorableNode(node) As of v1.2, moved to ReordererDomUtilities.js

Description

Determine if a node should be ignored by the iterator functions. A text node that is all whitespace and comment nodes should all be ignored.

Arguments

- node {Element}: representing the DOM1 Node interface

Return Value

Boolean: true if the node is a Text node that is all whitespace or a comment node, otherwise false.

Reference
See: http://developer.mozilla.org/En/Whitespace_in_the_DOM

fluid.dom.isWhitespaceNode(node) As of v1.2, moved to ReordererDomUtilities.js

Description

Determine whether a node's text content is entirely whitespace

Arguments

- node {Element}: a node implementing the CharacterData interface (i.e., a Text, Comment, or CDATASection node

Return Value

Boolean: true if all of the text content of node is whitespace, otherwise false.

Reference
See: http://developer.mozilla.org/En/Whitespace_in_the_DOM

fluid.dom.iterateDom(node, acceptor, allNodes)

Description

Walks the DOM, applying the specified acceptor function to each element. There is a special case for the acceptor, allowing for quick deletion of elements and their children.

Arguments

- node {Element}: a node to start walking the DOM from
- acceptor {Function}: the function to invoke with each DOM element. If the return value is "delete", the element in question will be deleted. If the return value is "stop" the iteration will be terminated
- New in v1.1: allNodes {Boolean}: use **true** to call acceptor on all nodes, rather than just element nodes (type 1)

Return Value

No Return Value

fluid.dom.iterateDom.DOM_BAIL_DEPTH

Description

This is provided as a work around for IE's circular DOM issue. This is the default max DOM depth in IE.

Constant Value

Constant: 256

Reference

http://msdn2.microsoft.com/en-us/library/ms761392(VS.85).aspx

RendererUtilities.js (new in v1.3)

New in v1.3: fluid.initRendererComponent()

Please see fluid.initRendererComponent in the new Documentation space.

On This Page

- fluid.allocateSimpleId(element)
- New in v1.3: fluid.arrayToHash()
- fluid.byId(id, dokkument)
- fluid.clear(target)
- fluid.COMPONENT_OPTIONS
- New in v1.3: fluid.computeNickName()
- fluid.container(containerSpec, fallible)
- fluid.copy(tocopy)
- fluid.createDomBinder(container, selectors)
- fluid.defaults()
- New in v1.3: fluid.demands()
- New in v1.3: fluid.each()
- fluid.emptySubcomponent(options)
- fluid.event.getEventFirer(unicast, preventable)
- New in v1.3: fluid.event.identifyListener()
- New in v1.3: fluid.expandOptions()
- New in v1.3: fluid.expect()
- fluid.expectFilledSelector(result, message)
- fluid.fail(message)
- fluid.fetchResources()
- New in v1.3: fluid.fetchResources.primeCacheFromResources()
- fluid.find(list, fn, deflt)
- fluid.findAncestor(element, test)
- fluid.findKeyInObject
- fluid.freshContainer(tocopy)
- fluid.formatMessage(messageString, args)
- new in v1.3: fluid.get(root, EL)
- fluid.getGlobalValue(path, env)
- fluid.getId(element)

- fluid.identity (element), fluid.dumpEl (element), fluid.renderTimestamp(element)
- New in v1.3: fluid.initDependent()
- New in v1.3: fluid.initDependents()
- fluid.initDomBinder (that)
- fluid.initLittleComponent(name, options)
- fluid.initSubcomponent(that, className, args)
- fluid.initSubcomponents(that, className, args)
- fluid.initView (componentName, container, userOptions)
- fluid.instantiateFirers (that, options)
- New in v1.3: fluid.invoke()
- fluid.invokeGlobalFunction (functionPath, args, environment)
- New in v1.3: fluid.iota()
- fluid.isArrayable (totest)
- New in v1.3: fluid.isMarker()
- fluid.isPrimitive (value)
- fluid.jById(id, dokkument)
- fluid.keyForValue (obj, value)
- fluid.log(str)
- fluid.merge (policy, target)
- fluid.mergeComponentOptions(that, componentName, userOptions)
- fluid.mergeListeners (events, listeners)
- fluid.messageLocator (messageBase)
- New in v1.3: fluid.messageResolver()
- fluid.model.composePath (prefix, suffix)
- New in v1.3: fluid.model.composeSegments()

- fluid.model.copyModel(target, source)
- fluid.model.getBeanValue(root, EL)
- fluid.model.getPenultimate(root, EL, environment, create)
- fluid.model.parseEL(EL)
- fluid.model.setBeanValue(root, EL, newValue)
- New in v1.3: fluid.model.resolvePathSegment()
- New in v1.3: fluid.model.transformWithRules()
- New in v1.3: fluid.prettyPrintJSON()
- New in v1.3: fluid.progressiveChecker()
- fluid.registerGlobal(functionPath, func, env), fluid.registerGlobalFunction(functionPath, func, env)
- fluid.registerNamespace(naimspace, env)
- fluid.remove_if(list, fn)
- New in v1.3: fluid.render()
- New in v1.3: fluid.renderer.createRendererFunction()
- New in v1.3: fluid.renderer.makeProtoExpander()
- New in v1.3: fluid.renderer.mergeComponents()
- New in v1.3: fluid.renderer.selectorsToCutpoints()
- new in v1.3: fluid.set(root, EL, newValue)
- fluid.setLogging(enabled)
- fluid.stringTemplate(template, values)
- fluid.transform(list)
- fluid.unwrap(obj)
- fluid.version
- fluid.wrap(obj)

- FluidDOMUtilities.js
    - fluid.dom.cleanseScripts(element) As of v1.2, moved to ReordererDomUtilities.js
    - fluid.dom.cleanseScripts.MARKER As of v1.2, moved to ReordererDomUtilities.js
    - fluid.dom.computeAbsolutePosition(element) As of v1.2, moved to ReordererDomUtilities.js
    - fluid.dom.getElementText(element)
    - fluid.dom.insertAfter(newChild, refChild) As of v1.2, moved to ReordererDomUtilities.js
    - fluid.dom.isContainer(container, containee)
    - fluid.dom.isIgnorableNode(node) As of v1.2, moved to ReordererDomUtilities.js
    - fluid.dom.isWhitespaceNode(node) As of v1.2, moved to ReordererDomUtilities.js
    - fluid.dom.iterateDom(node, acceptor, allNodes)
    - fluid.dom.iterateDom.DOM_BAIL_DEPTH
- RendererUtilities.js (new in v1.3)
    - New in v1.3: fluid.initRendererComponent()