fluid.initRendererComponent

This functionality is Sneak Peek status. This means that the **APIs may change**. We welcome your feedback, ideas, and code, but please use caution if you use this new functionality.

fluid.initRendererComponent(componentName, container, options)

An initialiation method that can be called as the first act of any component that uses the Infusion Renderer.
This function automatically initializes the Renderer in addition to merging user options with defaults,
attaching a DOM Binder to the instance, and configuring events.

```
fluid.initRendererComponent(componentName, container, options);
```

File name: `RendererUtilities.js`

Parameters

`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.
`options`	(Object) The configuration options for this component

Return Value

Object	A renderer-bearing component object.

See Also

- Renderer-bearing Components
- How To Use The Renderer
- fluid.render
- fluid.initView
- fluid.renderer.createRendererFunction

Options

As with any component initialization function, the `options` object will be merged with any defaults found for the `componentName`, and the result will be attached to the returned object. In general, options are specific to each component.

This renderer initialization function, however, recognizes certain particular options that will be used to configure the renderer. A component may wish to define defaults for some of these options. Others will be provided by the component user (e.g. `model`). The renderer-specific options are:

Unable to render {include} The included page could not be found.

Renderer Function Options

The `fluid.initRendererComponent` initializer will create a `render` function that will be attached to the `that` object. The `options` parameter includes an options called `rendererFnOptions` that can contain options that will control the behaviour of the renderer function. The following renderer function options are supported:

Name	Description	Values	Default
`cutpointGenerator`	A function that can be used to create a list of cutpoints.NOTE that this function will *only* be used if the `rendererOptions` option does not include a `cutpoints` entry.	function	`fluid.renderer.selectorsToCutpoints`
`expanderOptions`	Options that will be passed on to the prototree expander function. Note that if the `noexpand` option is `true`, these options will be ignored.	Object	none
`noexpand`	A flag that, if `true`, will prevent the prototree from undergoing any expansion. NOTE that this option will render any `expanderOptions` meaningless.	boolean	`false`
`rendererOptions`	Options that will be passed on to the Renderer itself. See fluid.render for more information	Object	none

templateSource	A string selector, DOM node or jQuery object containing the source template to be used for rendering by this function.	jQueryable	none; The container will be used as the source for the template if this option is not provided.

Renderer Options

The render function will use the Infusion Renderer to render the component. The Renderer itself can be configured using options, which can be provided using the rendererOptions option. The following renderer options are supported:

Field	Description	Type	In/Out
model	Perhaps the most important parameter, contains the "data model" to which value bindings expressed within the tree will be expressed.	free Object	Mostly In, but the supplied model may be written to later as a result of servicing user actions, especially if the parameter autoBind is supplied.
applier	a ChangeApplier object associated with the model	ChangeApplier	In
autoBind	If set, any user modification of fields with valuebindings s set will immediately be reflected in the current state of the supplied model	boolean	In
document	If set, will cause the rendered ids to be uniquified against the supplied document, rather than the current one	document	In
debugMode	If set, mismatches between template and component tree will be highlighted in an unreasonable garish pink colour	boolean	In
idMap	This map operates in conjunction with the identify decorator which may be attached to nodes in the component tree. Whilst rendering takes place, this map will fill up with a lookup from the supplied nickname to the finally rendered id	free Object	In/Out
messageLocator	Configures the lookup from (I18N) messages referenced in the component tree, to some source of a message bundle	function (key, args)->message	In
messageSource	Will construct a messageLocator from a raw bundle specification via a call to fluid.resolveMessageSource	MessageSource structure	In
renderRaw	Will XMLEncode the rendered markup before insertion into the document. Can be useful for debugging	boolean	In
cutpoints	This is properly a directive to the parser, rather than the renderer, but the options structure is shared. This contains a list of pairs of id, selector which will be used to impute an rsf:id structure onto a document, by means of matching the paired selector	Array of Cutpoint object	In

Example

```
// Declare defaults for the component
fluid.defaults("cspace.autocomplete.popup", {
    selectors: {
        addToPanel: ".csc-autocomplete-addToPanel",
        authorityItem: ".csc-autocomplete-authorityItem",
        noMatches: ".csc-autocomplete-noMatches",
        matches: ".csc-autocomplete-matches",
        matchItem: ".csc-autocomplete-matchItem",
        longestMatch: ".csc-autocomplete-longestMatch",
        addTermTo: ".csc-autocomplete-addTermTo"
        },
    repeatingSelectors: ["matchItem", "authorityItem"],
    resources: {
        template: {
            href: "html/AutocompleteAddPopup.html"
        }
    },
    ....
};

// Component creator function
cspace.autocomplete.popup = function(container, options) {
    var that = fluid.initRendererComponent("cspace.autocomplete.popup", container, options);
    ...
    return that;
};
```

This example uses fluid.initRendererComponent() to initialize a component called cspace.autocomplete.popup. The defaults for the component include defaults for several renderer-specific options: selectors, repeatingSelectors and resource.

The that object that is returned by the call to fluid.initRendererComponent() will include a render() function that can be used to render the data model provided by the implementor, using the provided selectors. The HTML file referenced by template.href will be retrieved and stored back in the template for use by the component.