Declarative this-ism in IoC

Introduction

Typically in Infusion, all functions and components adhere to a "that-ist" approach. This approach implies that the meaning of any function value remains the same, however the function is invoked. This is in contrast to the "this-ist" approach used frequently in JavaScript code, where the context of a function (the value of `this` within its body) depends on where the caller has stored it before invoking it. When working with Infusion, there are times when you will need to interact with a library and or function that requires a "this", which is a keyword typically intended to refer back to itself. The most common example would be using a jQuery element.

For more details on the differences between "that-ist" and "this-ist" approaches, as well as the reasoning behind why Infusion employs the former, see the About this and that blog post.

Structure

A this-ist function can be declaratively bound to invokers and listeners in an IoC tree and takes the following structure:

```
{
    "this": "{that}.dom.elm",
    "method": "click",
    "args: ["console.log"]
}
```

Property	Description
`"this"`	The "this" required by the function; the object that the function will be called on. Note that this property name must include the quotes (" "). e.g. a jQuery object
`method`	The name of the function to be called. e.g. a jQuery function like "click"
`args`	(optional) The argument or array of arguments to be passed into the function. This can include IoC References and expanders, in addition to strings, objects, booleans, etc.
`namespace`	(optional)

Examples

In the following example, a this-ist function is use to attach a listener to the `onCreate` events: The jQuery `click()` function of the button identified by the `"button"` selector is used to bind the component's `writeText()` method to the click event.

```
fluid.defaults("demo.hw", {
    gradeNames: ["fluid.viewComponent", "autoInit"],
    selectors: {
        button: "demo-hw-button",
        title: "demo-hw-title"
    },
    strings: {
        title: "Hello World!"
    },
    invokers: {
        // writes the supplied string to the title element
        writeText: {
            "this": "{that}.dom.title",
            "method": "text",
            "args": ["{that}.options.strings.title"]
        }
    },
    listeners: {
        // binds a click handler to the button element
        onCreate: {
            "this": "{that}.dom.button",
            "method": "click",
            "args": ["{that}.writeText"]
        }
    }
});
```

In the following example, taken from the Infusion Table of Contente comonent, this-ist functions are used to attach methods to the component object in the `invokers` option.

```
fluid.defaults("fluid.tableOfContents", {
    gradeNames: ["fluid.viewComponent", "autoInit"],
    ...
    invokers: {
        ...
        hide: {
            "this": "{that}.dom.tocContainer",
            "method": "hide"
        },
        show: {
            "this": "{that}.dom.tocContainer",
            "method": "show"
        }
    },
```

In the following example, taken from the Infusion Inline Edit component, a this-ist function is used to invoke the `init` function on the tinyMCE object associated with the component as a listener on the `onCreate` event.

```
fluid.defaults("fluid.inlineEdit.tinyMCE", {
    gradeNames: ["fluid.inlineEdit", "autoInit"],
    listeners: {
        onCreate: {
            "this": "tinyMCE",
            method: "init",
            namespace: "initTinyMCE",
            args: "{that}.options.tinyMCE"
        }
    },
    ...
});
```

In the following example, taken from the Infusion Pager component, a this-ist function is used to set the container's `role` attribute to "application" using the jQuery `attr()` function.

```
fluid.defaults("fluid.pager", {
    gradeNames: ["fluid.viewComponent", "autoInit"],
    ...
    listeners: {
        onCreate: {
            "this": "{that}.container",
            method: "attr",
            args: ["role", "application"]
        },
    ...
});
```