`fluid.function`.

The framework's built-in component grades

The Infusion Framework already contains several predefined component grades that normally form the initial building blocks for external components and grades. The following table describes these grades and how they relate to each other.

Grade Name	Description
`fluid.component` *Changed in version 2.5.3: this component has been added*	A plain `fluid.component` is the most basic component: it supports options merging with defaults (Components), as well as instantiating event firers based on default framework events (`onCreate`, `onDestroy`, `afterDestroy`) and events declared in the options (Tutorial - Creating Components). All Infusion components are derived from this grade, and in general all things not derived from this grade are non-components (e.g. plain functions, or model transformation transforms, etc.)
`fluid.modelComponent`	A *model* component is a component that additionally provides supports for a component's model, and operations on it (Tutorial - Model Components). These operations are mediated by a machine known as a `ChangeApplier` which is automatically constructed for a model component. As well as exposing a programmatic API, this also allows for declarative constraints and relationships to be enforced by means of the model relay system.
`fluid.viewComponent`	A *view* component is a `fluid.modelComponent` that is bound to a DOM container node, holds a DOM Binder and supports a view (Tutorial - View Components).
`fluid.rendererComponent`	A *renderer* component is a view component that also bears a renderer. There are additional features provided by this component grade specified on the Useful functions and events section of the Tutorial - Renderer Components page

Specifying Parent Grades

Function Grades

Edit on GitHub

Most grades represent Infusion components - these are derived from the base grade `fluid.component`. However, some grades describe plain JavaScript functions - these are derived from the base grade `fluid.function`. The purpose of the Fluid API call fluid.defaults could be understood as providing *metadata* about some element of the system. In the case of a full component grade, this metadata is sufficient to allow the framework to automatically construct the creator function for the component. In the case of a function grade which describes an already existing function with a global name, the metadata supplies hints to the user about how to call the function and its purpose.

Changed in version 3.4.10: this paragraph includes new content

Registering a global function

A global function is registered within infusion at a stable place in its global namespace by working with the core API functions fluid.registerNamespace and fluid.setGlobalValue - in practice the latter is rarely used, in favour of directly setting members on namespace objects.

If you are working in the browser, the global object (traditionally named `window`) coincides with Fluid's global object (which can be retrieved from `fluid.global` - assuming that you have already managed to resolve the `fluid` object itself). If you are working in node.js, you need to make calls to