Options Merging

The framework merges a component's defaults with the options specified by the user at runtime. Default options for a component are registered by the component developer using the `fluid.defaults()` function. The framework combines these values with those supplied by the user to the component creator function.

On this Page

Left unchecked, this options merging code becomes unruly and verbose. The vast majority of merging can be taken care of by a simple call to a standard "object merging function" such as `jQuery.extend` - however, there are a few cases where this behaviour is inappropriate, and where manual merging code might still be required.

The Fluid component API therefore contains a specialised merging routine, `fluid.merge`, which allows the user to specify an (optional) more fine-grained *merge policy object*, which allows detailed behaviour during options merging to still be specified by a declarative strategy.

`fluid.merge` signature

The signature to `fluid.merge` is

```
fluid.merge(policy, target, source-1, ... source-n);
```

where `policy` is the *merge policy object* (may be empty), `target` is the object to (destructively) be the target of the merge operation, and `source-1` through `source-n` are a list of a number of "source" options structure from which the merge is to be performed.

`fluid.merge` and `initView`

Typically the component implementor will not need to call `fluid.merge` manually, since it is automatically invoked as part of the operation of the standard `fluid.initView` initialisation call. `fluid.initView` will automatically look for a member of the default options registered for the component type (via `fluid.defaults`) named `mergePolicy`, and if found, apply this policy to merge the `userOptions` supplied with the remainder of the default options.

Therefore, to take advantage of a fine-grained merge policy, the component implementor needs only to understand the structure of the merge policy object, and add it to their overall default options registered with `fluid.defaults`.

Structure of the Merge Policy Object

The merge policy object is a hash of keys, which represent *EL paths* into the target object, onto values which represent a policy.

By default if there is no policy specified, a deep merge is done. Everything in the source objects are copied over the target object. Anything that existed in the target but was not present in any of the source objects will continue to be present in the resulting merged object.

The following are the policy types that are supported, determined by the key's value in the policy object.

Policy value	Description
`"replace"`	If any value is found at the EL path in a "source" object, the value held at the "target" path will be cleared (that is, blasted back to an empty state with no properties) before the values are copied.
New in v1.3: `"preserve"`	If any value is found at the EL path in a "source" object, the source object will be preserved i.e. the object will not be copied, but directly used. If any defaults exist, those will be merged into the source object.
New in v1.4: `"noexpand"`	
New in v1.4: `"nomerge"`	
`"reverse"`	This is scheduled for deprecation

Any other string value	The value will be interpreted as an EL path into source object. If after the merge operation, no value is present at the key EL value in the target object, this (right-hand) value will be interpreted as an EL path into the source, and the value copied from this EL into the key EL into the target.
function	Rather than perform any merge operation, the key will be interpreted as a function which is expected to accept two arguments (target, source) representing the sub-objects which are about to enter merging. The function will be invoked on these arguments and no further operations will be performed on this subtree.