Inline Edit Functions

Functions

These functions are defined on the central `that` object returned from the inlineEdit construction function - for example with

```
var that = fluid.inlineEdit(componentContainer, options);
```

```
that.edit();
```

Switches the component into edit mode. The events `onBeginEdit` and `afterBeginEdit` will fire.

```
that.finish();
```

Switches the component out of edit mode into display mode, updating the displayed text with the current content of the edit field. The events `onFinishEdi` and `afterFinishEdit` will fire. If the model value has changed, there will be a call to `modelUpdated` in between these calls.

```
that.cancel();
```

Cancels the in-progress edit and switches back to view mode.

```
that.isEditing();
```

Determines if the component is currently in edit mode: Returns true if edit mode is shown, false if view mode is shown.

```
that.refreshView(source);
```

Updates the state of the inline editor in the DOM, based on changes that may have happened to the model.

```
that.tooltipEnabled();
```

Returns a boolean indicating whether or not the tooltip is enabled.

```
/**
  * Pushes external changes to the model into the inline editor, refreshing its
  * rendering in the DOM. The modelChanged event will fire.
  *
  * @param {String} newValue The bare value of the model, that is, the string being edited
  * @param {Object} source An optional "source" (perhaps a DOM element) which triggered this event
  */
that.updateModelValue(newValue, source);
```

Updates the component's internal representation of the text to a new value. If the value differs from the existing value, the `modelChanged` event will fire and the component will be re-rendered.

```
/**
  * Pushes external changes to the model into the inline editor, refreshing its
  * rendering in the DOM. The modelChanged event will fire.
  *
  * @param {Object} newValue The full value of the new model, that is, a model object which
  *      contains the editable value as the element named "value"
  * @param {Object} source An optional "source" (perhaps a DOM element) which triggered this event
  */
that.updateModel(newValue, source);
```

Similar to `updateModelValue`, only accepts specification of the overall model object (housing the editable value under the key `value`).

```
that.model
```

Not a function, but a data structure. This directly represents the "model" or state of the editable component. External users should consider this structure as read-only, and only make modifications through the `updateModel` call above.

Options

The following options to the creator functions can be used to customize the behaviour of the Inline Edit component: