Creating Panels

This article describes how to use the Infusion Preferences Framework to create Panels, user interfaces that display adjusters, or controls, allowing users to modify their settings for various preferences. This is part of the Tutorial - Creating a Preferences Editor Using the Preferences Framework.

Overview

On This Page

See Also

- Understanding Infusion Components
- Component Grades
- IoC - Inversion of Control
- Renderer
- Infusion Framework Best Practices

In the Preferences Framework, "Panels" are Infusion renderer components that present adjusters, or controls to the user to allow them to adjust the preference settings. For more information about Infusion Components, see Understanding Infusion Components and other pages linked to in the "See Also" panel on this page.

The configuration information used to define a preferences panel must include certain required information:

- the `fluid.prefs.panel` and `autoInit` grades (provided by the Framework)
- a Preference Map (see below)
- a renderer proto-tree or `produceTree` function
- selectors for rendering the controls, labels, etc
- any other information required by the panel

PreferenceMap

Each panel and enactor defines a "preference map," which map the information in the Primary Schema into your Panel. The preference map is used to copy the default preference value from the primary schema into the Panel's model. It can also be used to copy any other necessary information from the primary schema into the Panel, such as enumerations, minimums and maximums, etc. The values can be mapped to any path in the Panels options, and then they can be accessed through those paths.

Format

```
preferenceMap: {
    <key from primary schema>: {
        <path in panel's options where value should be held>: <key in primary schema where value held>,
        <path in panel's options where value should be held>: <key in primary schema where value held>,
                ... any number of the above, as required ...
    },
    ... any number of the above, as required ...
}
```

Examples

```
fluid.defaults("fluid.prefs.panel.textSize", {
    gradeNames: ["fluid.prefs.panel", "autoInit"],
    preferenceMap: {
        "fluid.prefs.textSize": {
            "model.value": "default",
            "range.min": "minimum",
            "range.max": "maximum"
        }
    },
    ....
```

```
fluid.defaults("fluid.prefs.panel.textFont", {
    gradeNames: ["fluid.prefs.panel", "autoInit"],
    preferenceMap: {
        "fluid.prefs.textFont": {
            "model.value": "default",
            "controlValues.textFont": "enum"
        }
    },
    ....
```

```
fluid.defaults("fluid.videoPlayer.panels.captionsSettings", {
    gradeNames: ["fluid.videoPlayer.panels.mediaSettings", "autoInit"],
    preferenceMap: {
        "fluid.videoPlayer.captions": {
            "model.show": "default"
        },
        "fluid.videoPlayer.captionLanguage": {
            "model.language": "default"
        }
    },
    ....
```

Example: 'line spacing' preference panel

```
fluid.defaults("fluid.prefs.panel.lineSpace", {
    gradeNames: ["fluid.prefs.panel", "autoInit"],
    preferenceMap: {
        "fluid.prefs.lineSpace": {
            "model.value": "default",
            "range.min": "minimum",
            "range.max": "maximum"
        }
    },
    range: {
        min: 1,
        max: 2
    },
    selectors: {
        lineSpace: ".flc-prefsEditor-line-space",
        label: ".flc-prefsEditor-line-space-label",
        narrowIcon: ".flc-prefsEditor-line-space-narrowIcon",
        wideIcon: ".flc-prefsEditor-line-space-wideIcon",
        multiplier: ".flc-prefsEditor-multiplier"
    },
    protoTree: {
        label: {messagekey: "lineSpaceLabel"},
        narrowIcon: {messagekey: "lineSpaceNarrowIcon"},
        wideIcon: {messagekey: "lineSpaceWideIcon"},
        multiplier: {messagekey: "multiplier"},
        lineSpace: {
            decorators: {
                type: "fluid",
                func: "fluid.prefs.textfieldSlider"
            }
        }
    },
    sliderOptions: {
        orientation: "horizontal",
        step: 0.1,
        range: "min"
    }
});
```

Example: Video Player caption preferences panel

```
fluid.defaults("fluid.videoPlayer.panels.captionSettings", {
    gradeNames: ["fluid.prefs.panel", "autoInit"],
    preferenceMap: {
        "fluid.videoPlayer.captions": {
            // the key is the internal model path, the value is the path into the schema
            "model.show": "default"
        },
        "fluid.videoPlayer.captionLanguage": {
            "model.language": "default"
        }
    },
    model: {
        show: false,
        language: "en"
    },
    listeners: {
        onCreate: "fluid.videoPlayer.panels.captionSettings.toggleLanguageOnShow"
    },
    strings: {
        language: ["English", "French"]
    },
    controlValues: {
        language: ["en", "fr"]
    },
    styles: {
        icon: "fl-icon"
    },
    selectors: {
        label: ".flc-videoPlayer-media-label",
        show: ".flc-videoPlayer-media",
        choiceLabel: ".flc-videoPlayer-media-choice-label",
        language: ".flc-videoPlayer-media-language"
    },
    produceTree: "fluid.videoPlayer.panels.captionSettings.produceTree"
});
```

Back to Tutorial - Creating a Preferences Editor Using the Preferences Framework