Widgetset configuration concretes

Parent concept page describes how we should reorganize the widgetset to make it configurable and open for extension. This page highlights the concrete implementation choices (e.g. naming, packaging) that are made to fulfill the concept.

Configuration

magnolia.properties

New entries in magnolia-empty-webapp/src/main/webapp/WEB-INF/config/default/magnolia.properties:

magnolia.properties

```
# Change to point at your custom Vaadin widgetset and theme
# Your widgetset should always inherit magnolia's default widgetset (info.magnolia.ui.vaadin.gwt.
MagnoliaWidgetSet)
# Your theme should always include magnolia's default theme (admincentral)
magnolia.ui.vaadin.widgetset=info.magnolia.ui.vaadin.gwt.MagnoliaWidgetSet
magnolia.ui.vaadin.theme=admincentral
```

AdmincentralUIProvider

Package: *info.magnolia.ui.admincentral*

This is a custom Vaadin UIProvider. It gets the *MagnoliaConfigurationProperties* via injection, and checks for property values when queried for widgetset or theme. Vaadin's DefaultUIProvider can only resolve widgetsets and themes either from servlet parameters or via annotation.

Most magnolia **property keys** are defined as constants in the `SystemProperty` deprecated class. For these new properties, we keep the keys where we use them, i.e. in the AdmincentralUIProvider.

AdmincentralUIProvider.java

```
public static final String WIDGETSET_PROPERTY_KEY = "magnolia.ui.vaadin.widgetset";
public static final String THEME_PROPERTY_KEY = "magnolia.ui.vaadin.theme";
```

It also holds **fallback default values** in case widgetset and theme cannot be determined:

AdmincentralUIProvider.java

```
public static final String DEFAULT_WIDGETSET = "info.magnolia.ui.vaadin.gwt.MagnoliaWidgetSet";
public static final String DEFAULT_THEME = "admincentral";
```

Custom widgetset

1. Widgetset fundaments

In order to add custom widgets or Vaadin addons to your Magnolia app, you need to provide a new Widgetset.

Like for any GWT application, there can be only one Widgetset, therefore it should inherit all the Widgetsets you intend to use; in particular it must always inherit Magnolia's default widgetset (identified as info.magnolia.ui.vaadin.gwt.MagnoliaWidgetSet).

Finally, a widgetset must always be GWT-compiled before you run your Web application, and it will have to be recompiled every time the client-side code changes, or when you add that fancy Vaadin addon.

For further information about Vaadin client-side development, or about ways to compile the Widgetset,GWT compilation, please refer to the Book of Vaadin, or the GWT project website.

2. Using a custom widgetset in Magnolia

Once your widgetset is compiled, you may now tell magnolia to use it by editing your webapp's magnolia properties, and setting the key *magnolia.ui.vaadin.widgetset*.

The expected value is a widgetset qualified name as Vaadin expects it, without the .gwt.xml extension, e.g. some.vaadin.package.SomeWidgetset.

Custom app styles

1. Vaadin themes

Module styles are provided as Vaadin themes, generally using Sass.

Like for any Vaadin application, there can be only one Vaadin theme per Vaadin UI (i.e. Magnolia's AdmincentralUI). Therefore it should include all the other themes you intend to use; in particular it must always include Magnolia's default *admincentral* theme.

Sass themes offer the advantage of being composed from several other Vaadin themes, through Sass mixins. They are ultimately compiled and served as one big chunk of CSS.

For more information on Vaadin themes or Sass, you may refer to the Book of Vaadin or the Sass website.

2. Using custom styles in your Magnolia app

You can already use custom styles in your Magnolia app, by adding a theme property to your app descriptor, see App theme. However in some cases, you need the styles to be loaded without starting the app (e.g. custom dialog, custom message view).

In that case, once you have your Sass theme ready (includes the *admincentral*mixin and your custom styles), you may now tell magnolia to use it by editing your webapp's magnolia properties, and setting the key *magnolia.ui.vaadin.theme*.

The expected value is a theme name as Vaadin expects it, i.e. the name of the theme folder under src/main/resources/VAADIN/themes.