Config By X - File format discussions

See: Unconference 2014: ConfigByFile / ConfigByCode / Developer Sync Module

Purpose

Goals

In a nutshell, files containing configuration should be readable, editable by hand, and easy to merge.

Proposal

An XML file with each attribute listed on a separate line for readability.

Benefits of XML format:

- People are very familiar with the format. Frontend developers are familiar with it via HTML.
- There is a mature schema system for it. http://en.wikipedia.org/wiki/XML_schema (There is one for Json as well, but I didnt see any tools used it in a helpful way.)
- Tooling is sophisticated
  - Every editor supports syntax hilighting out of the box.
  - Intellij & Eclipse support autosuggest, autocomplete, error hilighting based on a linked schema file and additionally sophisticated code formatting, code folding, validation and outline view.
- Readability is decent.

Proposed mapping to xml:

- Each node in jcr maps to an xml node.
- Each property in jcr maps to an xml attribute.
- The name of the xml node is often the jcr node name, but in "collections" the xml node name should maybe become something that can be specified in the schema.
  For example in jcr the field nodes in the tab of a form have names like "email" and "city", but we might want the xml node name for all fields to be "field".

The simpler the files can be, the easier they are to work with. We should try to remove as many nodes and attributes as possible - such as the metadata, and where possible nodetype and uuid. Intelligent defaults would allow us to further reduce configuration (for example: all edit subapps and dialogs have commit & cancel actions by default, all form fields are textfields by default.)

A few examples based on this proposal:

A simple template: template-stkSearchResult.xml

A complicated template: template-stkArticle.xml

An App, w/o actionbar: app-contacts.xml (+ an XML schema for it if you want to test IDE support app.xsd)

Research

Formats to consider:

- xml
- json
- yaml

Comparing file formats

Please see zip attachment for a set of files with alternative formatting techniques. (for convenience a few files are attached themselves to this page - but the complete set is in the zip file.)

Compare ConfigFile formats.zip

The node being compared is the dialog for stkHTML STK component.
You can find it in AdminCentral "config" workspace @:
/modules/standard-templating-kit/dialogs/components/content/stkHTML

Slim versions

In the "slim" versions, I've removed all metadata - but probably removed too much. Final versions should include jcr:contentType and perhaps UUID.
But I have tried to at least be consistant across the different versions.

Most of the files are based on existing ways of outputting configuration to files.
The "dialog-json-*" and "dialog-whitespace slim.txt" files are pure inventions.

"dialog-whitespace slim.txt" could be implemented in YAML. http://www.yaml.org/start.html

Tooling for JSON & YAML

There is a spec for JSON schema, which might be handy for enforcing the json structure to be correct - (could be a jshint build process step):

http://json-schema.org/documentation.html

JSON Schema based validation in Eclipse: https://github.com/sabina-jung/JSON-Schema-Validation-Eclipse

YAML would probably be the cleanest for diffs & merging.

YAML Eclipse tooling: https://code.google.com/p/yedit/

YAML Indenting Intellij: http://www.jetbrains.com/idea/webhelp/code-style-yaml.html