2017-08-31 Workshop: Content Types follow-up

31 Aug 2017

⚠ DEV-619 - Jira project doesn't exist or you don't have permission to view it.

Attendees

- Mikaël Geljić
- Ilgun Ilgun
- Philip Mundt
- Aleksandr Pchelintcev

Outcome

We aimed at simplifying the overall effort; we envisioned a production-ready foundation, to build further features upon (keeping these compartmented to follow-up stories). We also crafted proto-defs of a content-type definition, its related app descriptor and endpoint definition.

Discussion log

- Terminology still accurate
- proposal of a `ModelDefinition` in between `ContentTypeDefinition` and `PropertyDefinitions`
  - Keep sub-models for later [1]
  - Support top-level basic cases first
- Use datasources first as a configuration shim
  - **no DS Java API involved**
  - no contract of what a Java Datasource is atm
- JCR datasource definition
  - take over **typical structure from JcrContentConnectorDefinition** to start with
- Practically perform an *Inversion of Control* from apps (also aka *Inversion of Configuration*)
  - instead of config' everything in-place
  - injecting CT model from which fields are derived
- Save implementations piggy-backing on CTs for later
  - e.g. new apps/form components on Vaadin 8
  - maybe w/ impl provision via SPI
- Can do dialog/form def improvements on the side as well (e.g. tabs)

App generation

- Opt-in for app generation
- require a proper apps/<foo>.yaml file
- **minimal app-descriptor** with reference to CT
  - A. via string ref, or yaml tag !autogenerate (marker/construct/inflater)
    - decoration of a prototype?
  - B. or without proxy definition from presenters?
    - but good hard-coded defaults
    - potentially harder to find paths for decorations?
  - backwards-compatibility?

- [ ] Ticket the 2 approaches (short timebox, avoid competition and stick all together in the loop)

Type aliases

Would benefit from type aliases for "datasource types"

- type: jackrabbit (instead of class: `info.magnolia.*.JackrabbitDatasourceDefinition`)
- would need a DatasourceRegistry, maybe

- => More aliases are crucial to remove boilerplate (datasource types, value types, validators)

Capabilities

Top-most bloat from content app config is actions / action bar config

- omit it for brevity [2]
- Address action mess / composing in a parallel effort
- using "generic" action-names / verbs, (rename, edit)
  research all apps and express those capabilities, default groups of actions
    - bulk-action, single-item action
    - CRUD
    - Publication
    - Import/Export
    - Version actions

Output filtering

Which fields you get from the endpoint

- with JCR delivery you get whatever properties exist under that node
- with CT delivery you get whatever properties are defined in the model
- consider detail levels later, there's a dedicated story for that [3]

Data source initialization

- Auto-creation of workspace and registration of namespaces/nodetypes
    - we still a DS Java class doing it for us
    - or autoCreatorClass?
- Superuser ACLs
- Subscription
  DS optional interface handling lifecycle (start/stop)
    - maybe also for autocreator
    - void/marker class, and implement additional interfaces you need
    - [ ] Ticket that

Recap of features left aside

- [1] sub-models
- [2] actions composition & capabilities
- [3] output filtering
- [4] type aliases
- [5] new app/form components

Proto-defs

/basel/contentTypes/contact(s).yaml

```
datasource:
  class: i.m.JackrabbitDataSourceDefinition       # could be omitted, resolves to magnolia/Jackrabbit by
default
  workspace: contacts                             # if omitted, same as CT name
  rootPath: /                                     # defaults to "/"
  nodeTypes:                                      # defaults to mgnl:content (TODO validate mixins from mgnl:
content)
    - name: mgnl:content                          # or to the extreme nt:unstructured
      icon: icon-user-public
  autoCreate: true

model: # can still use alternative map syntax
  - name: jcrName
    readOnly: true
  - name: salutation
  - name: firstName
    required: true
  - name: lastName
    required: true
  - name: photo
    type: binary (data-driven) OR upload (field-driven)
  - name: email
    validators:
      - class: info.magnolia....EmailValidatorDefinition (or alias validators as well)
  - name: dateOfBirth
    type: date
  - name: isPhilip
    type: boolean (data-driven) OR checkbox (field-driven)
    defaultValue: false
```

/basel/apps/contacts.yaml

```
icon: icon-contacts-app                           # defaults to a default icon
label: Contacts                                   # defaults to capitalized app name
class: info.magnolia.ui.contentapp.ContentAppDefinition       # defaults to a content-app?
contentType: contacts                             # defaults to app name
```

Rename `AppDescriptor` to `AppDefinition` for consistency (and AppDesc extends from AppDef)

/basel/endpoints/contacts.yaml

```
class: info.magnolia.rest.delivery.DeliveryEndpointDefinition
contentType: contacts                             # defaults to endpoint name
```

Next steps

- Quick cleanup of the datasource Java API on the branch (removal)
- Tickets for:
  - Ilgun Ilgun: DS initialization: ~~MGNLCT-13~~ - Register namespaces for jackrabbit datasource CLOSED
  - Content App provisioning

- Aleksandr Pchelintcev: full-fledged definition via builder:

  DEV-643 - Jira project doesn't exist or you don't have permission to view it.

- Mikaël Geljić: alleviate need for definitions

  DEV-644 - Jira project doesn't exist or you don't have permission to view it.