KVD

Configure Data Layer model nodes

The KVD app supports ctrlX Data Layer models in YAML format (*.yaml) to populate model node instances from. The models have to follow the *ctrlX Data Layer schema definition*.

Please see the sample models in *kvd/models* folder for further details.

Example: Model

```
$schema: datalayer.v1.0.schema.json
name: snake
version: 0.0.1
elements:
    weight:
        description: The weight of the snake.
        nodeClass: Variable
        mode: read-write
        value: 42
        typeSchema:
            type: int32
    venomous:
        description: The snake bite is venomous.
        nodeClass: Variable
        mode: read-write
        value: true
        typeSchema:
            type: bool8
```

Model interpretion and limitations

- The KVD app tries to interprete the models lazy without validating it's syntax for correctness against it's schema definition.
- Unknown or not useful configurations simply were ommitted (e.g. *itemsModel* of a collection, ...).

Upload models

Upload your ctrlX Data Layer models (*.yaml) to appdata location:

```
kvd/models
```

Configure an model instance

Add the node instances to the nodes configuration file *nodes.json*:

1. Set the *model* of the node instance to the file name of your model e.g. *animals.v1.0.yaml*.
2. Set the *nodeClass* of the *metadata* to *Resource*.

Example

```
{
    "address": "samples/kvd/models/zoo/animals",
    "model": "animals.v1.0.yaml",
    "metadata": {
        "nodeClass": "Resource",
        "description": "This is a animals model instance node."
    }
},
{
    "address": "samples/kvd/models/wildlife/animals",
    "model": "animals.v1.0.yaml",
    "metadata": {
        "nodeClass": "Resource",
        "description": "This is a animals model instance node."
    }
},
...
```

Read-only nodes

Configure a node's value to be *read-only* (e.g. any not writable constant value) with it's *mode* set to *read-only* instead of *read-write*.

Example: read-only node

```
...
elements:
    my_constant:
        description: A constant value.
        nodeClass: Variable
        mode: read-only
        value: 42
        typeSchema:
            type: int32
...
```

Referencing other models

Recursive model hierarchies references are supported, allowing composition of objects using a class oriented approach to configure a more complex node tree.

Ensure all model files to located in the *kvd/models* directory.

Use the *$ref* keyword to insert nodes of a child model to a given parent node.

Example: model references

Our main model *animals.v1.0.yaml*:

```
$schema: datalayer.v1.0.schema.json
name: animals
version: 0.0.1
elements:
    count:
        description: The number of animals.
        nodeClass: Variable
        mode: read-only
        value: 42
        typeSchema:
            type: string
    reptiles:
        description: The reptiles.
        $ref: reptiles.v1.0.yaml
    ...
```

With it's child model *reptiles.v1.0.yaml*:

```
$schema: datalayer.v1.0.schema.json
name: reptiles
version: 0.0.1
elements:
    snake:
        description: The snake.
        $ref: snake.v1.0.yaml
    ...
```

With it's child model *snake.v1.0.yaml*:

```
$schema: datalayer.v1.0.schema.json
name: snake
version: 0.0.1
elements:
    weight:
        description: The weight of the snake.
        nodeClass: Variable
        mode: read-write
        value: 42
        typeSchema:
            type: int32
    venomous:
        description: The snake bite is venomous.
        nodeClass: Variable
        mode: read-write
        value: true
        typeSchema:
            type: bool8
```