

Changes in Roman configs and CLI

Pinja Pessi

23.8.2019

What is Roman?

- used for compiling course material from RST files
- reads steps from a config file and runs them in Docker
- two config files: project config for steps and global config for backend settings
- jsonschema validator is used to validate the config files

What have I been working on?

- schema for project config
- steps have names, can be used for referring to steps in commands
- changes in build output: shows step names (instead of indexes) and has colours
- several new CLI commands, mostly focused on editing the configs
- environment variables with some Bash-style parameter substitution
- improved validation error rendering
- some other stuff: tests, clean build, command for deleting Docker containers, cache file for Docker image download times, env variable loading order fixed, renamed `course.yml` to `roman.yml`

Changes to roman.yml (previously course.yml)

```
version: '2.0'
```

```
environment: ← project can have environment
```

- VAR1: hello ← env is now a list
- VAR2: \${VAR1}! ← Bash-style parameter substitution in env

```
steps:
```

- img: apluslms/compile-rst
 - cmd: make touchrst html
 - mnt: /compile
 - env:
 - STATIC_CONTENT_HOST: http://localhost:8080/static/default
- name: hello ← **step names**
 - img: hello-world

Build output

new cli stuff

`build`: builds the project

- `--clean`: deletes old build files before building
- `-s, --steps`: select which steps to build
- `--no-color`: print build output without colors

`prepare`: only does the preparation phase of building (downloads images)

- `-f, --force`: download all images
- `--steps STEPS [STEPS ...]`: same as `--steps` in `build`

`init`: creates new project config in current folder

commands for editing configs: `config` and `step` (including env specific commands)

Environment variables

- environment variables can be defined in global settings, project settings and individual steps
- environments from these three are combined
- steps overwrite project, project overwrites global
- environments are lists, and variables are expanded in order
- formats for list items:
 - a: b
 - a=b
 - name: a
value: b
 - name: a
unset: true

```
var1: 'aaa'
```

basic substitution:

```
${var1}! → 'aaa!'
```

default value:

```
${var1:-b} → 'aaa'
```

```
${var0:-b} → 'b'
```

alternate value:

```
${var1:+b} → 'b'
```

```
${var0:+b} → ''
```

pattern substitution:

```
${var1/a/b} → 'baa'
```

```
${var1//a/b} → 'bbb'
```

nesting

```
${var0:-${var1}} → 'aaa'
```

Env example:

roman_settings.yml:

```
version: '1.0'
```

```
environment:
```

- VAR1: hello
- VAR2: a

roman.yml:

```
version: '2.0'
```

```
environment:
```

- VAR1: \${VAR1}!
- VAR2: b

```
steps:
```

- name: hello
img: hello-world
env:
 - greeting: \${VAR1/e/a}
 - name: VAR1unset: True

resulting env:

```
VAR2: b  
greeting: hallo!
```


Validation errors

- validation errors are caused when a file doesn't follow the schema
- Roman renders the errors nicely so that error location is visible:

```
0: version: '1.0'
1:
2: docker:
3:   timeout: a
           ^
           validator: type
           description: default timeout for API calls
           type: integer
           title: docker API timeout
           minimum: 0
           exclusiveMinimum: True
Validation error: 'a' is not of type 'integer'
```

Validation errors in lists caused errors in the error render function:

Traceback (most recent call last):

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms-yamlidator/apluslms-yamlidator/document.py", line 267, in validate
    validator.validate(self._data)
```

```
File "/u/97/pessip1/unix/.local/lib/python3.5/site-packages/jsonschema/validators.py", line 348, in validate
    raise error
```

jsonschema.exceptions.ValidationError: 123 is not valid under any of the given schemas

Failed validating 'oneOf' in schema['properties']['steps']['items']:

```
  {'oneOf': [{'$ref': '#/definitions/image'},
             {'$ref': '#/definitions/step'}]}
```

On instance['steps'][0]:

```
  123
```

During handling of the above exception, another exception occurred:

Traceback (most recent call last):

```
File "/u/97/pessip1/unix/.local/bin/roman", line 11, in <module>
```

```
    load_entry_point('apluslms-roman', 'console_scripts', 'roman')()
```

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms_roman/cli.py", line 328, in main
```

```
    exit(context.run())
```

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms_roman/cli.py", line 50, in run
```

```
    return self.action(self)
```

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms_roman/cli.py", line 473, in config_print_action
```

```
    yaml_dump(get_config(context)._data, stdout)
```

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms_roman/cli.py", line 349, in get_config
```

```
    return ProjectConfig.find_from(getcwd())
```

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms_roman/configuration.py", line 48, in find_from
```

```
    return cls.load(config)
```

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms-yamlidator/apluslms-yamlidator/document.py", line 206, in load
```

```
    document = container.get_latest(max_version=version)
```

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms-yamlidator/apluslms-yamlidator/document.py", line 136, in get_latest
```

```
    return self._getitem(self._versions[version], validate=validate)
```

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms-yamlidator/apluslms-yamlidator/document.py", line 111, in _getitem
```

```
    document.validate()
```

```
File "/u/97/pessip1/data/Documents/aplus/roman/apluslms-yamlidator/apluslms-yamlidator/document.py", line 275, in validate
```

```
    ln, cn = dict_.lc.value(key)
```

AttributeError: 'list' object has no attribute 'lc'

'oneOf' in schemas combined with using refs resulted in unclear messages that don't really tell anything:

```
jsonschema.exceptions.ValidationError: ChangesDict(data=OrderedDict([('name', 'hello')]),
default=OrderedDict()) is not valid under any of the given schemas
```

```
Failed validating 'oneOf' in schema['properties']['steps']['items']:
```

```
  {'oneOf': [{'$ref': '#/definitions/image'},
             {'$ref': '#/definitions/step'}]}
```

```
On instance['steps'][0]:
```

```
  ChangesDict(data=OrderedDict([('name', 'hello')]), default=OrderedDict())
```

printing errors separately: some improvement, but still has unnecessary data:

There were several errors:

```
Error at steps[0]:
```

```
  ValidationError: ChangesDict(data=OrderedDict([('name', 'hello')]),
default=OrderedDict()) is not of type 'string'
```

```
Error at steps[0]:
```

```
  ValidationError: 'img' is a required property
```

```
ValidationError: ChangesDict(data=OrderedDict([('name', 'hello')]), default=OrderedDict())
is not valid under any of the given schemas
```

earlier:

```
items:
  oneOf:
    - $ref: "#/definitions/image"
    - $ref: "#/definitions/step"
```

current:

```
stepitem:
  if:
    type: string
  then:
    $ref: "#/definitions/image"
  else:
    if:
      type: object
    then:
      $ref: "#/definitions/step"
    else:
      type: [string, object]
```

```
0: version: '2.0'
1:
2: steps:
3:   - name: hello
      ^
      validator: required
      properties: ['env', 'cmd', 'mnt', 'img', 'name']
      additionalProperties: False
      required: ['img']
      type: object
```

ValidationError: 'img' is a required property

```
0: version: '2.0'
1:
2: steps:
3:   - 123
      ^
      validator: type
      type: ['string', 'object']
```

ValidationError: 123 is not of type 'string', 'object'