



adaptTo()

APACHE SLING & FRIENDS TECH MEETUP

BERLIN, 22-24 SEPTEMBER 2014

Apache Sling Generic Validation Framework



Radu Cotescu

ASF committer, Sling contributor

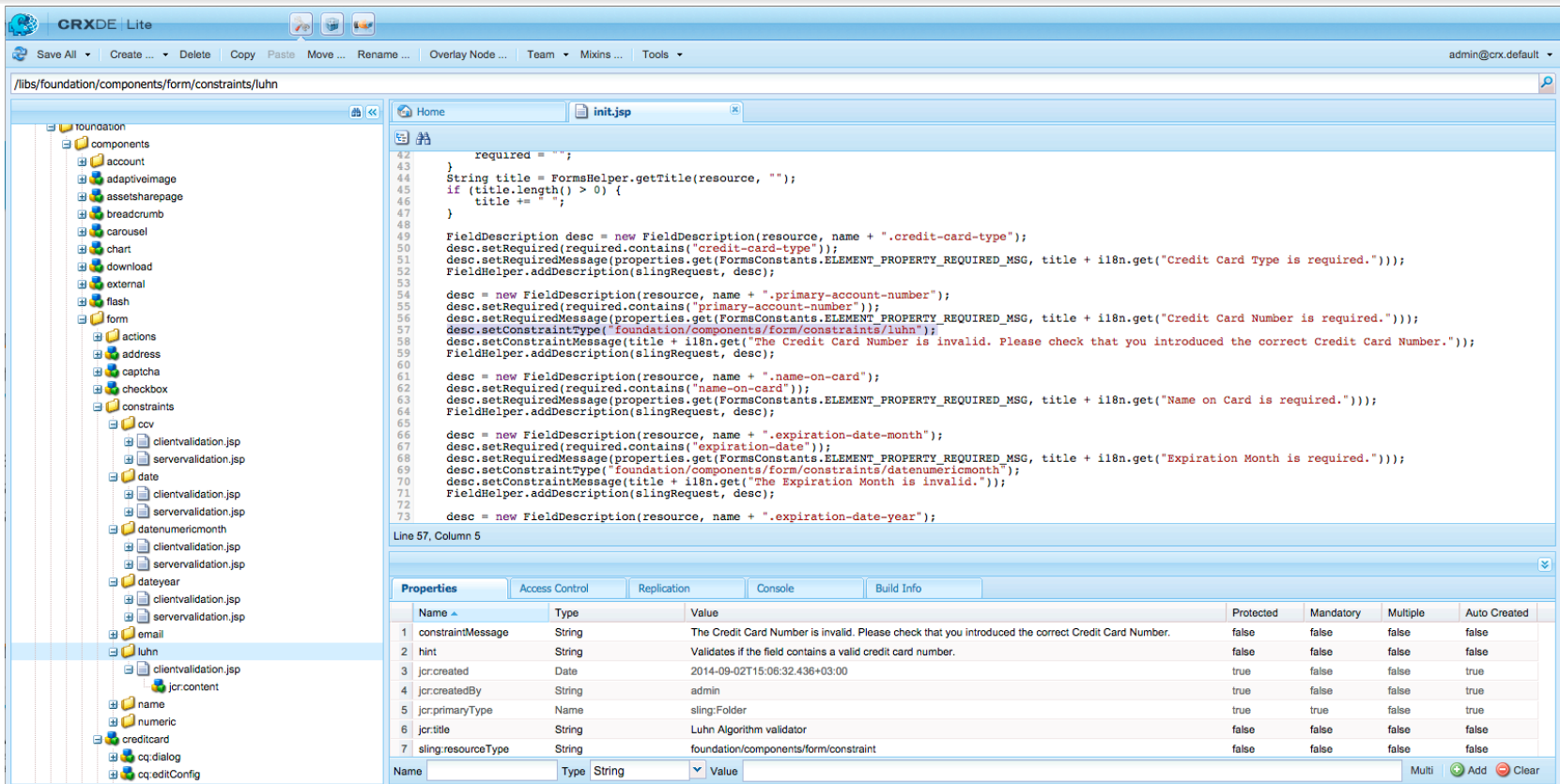
Computer Scientist @ Adobe Systems

@raducotescu - radu@apache.org



The most common web application security weakness is the failure to properly validate input from the client or environment. [1]

A Bit of History



The screenshot shows the CRXDE Lite interface. The left sidebar displays a tree view of the repository structure, with the path `/libs/foundation/components/form/constraints/luhn` selected. The main editor area shows the `init.jsp` file with the following code:

```

42     required = "";
43     }
44     String title = FormsHelper.getTitle(resource, "");
45     if (title.length() > 0) {
46         title += " ";
47     }
48
49     FieldDescription desc = new FieldDescription(resource, name + ".credit-card-type");
50     desc.setRequired(required.contains("credit-card-type"));
51     desc.setRequiredMessage(properties.get(FormsConstants.ELEMENT_PROPERTY_REQUIRED_MSG, title + i18n.get("Credit Card Type is required.")));
52     FieldHelper.addDescription(slingRequest, desc);
53
54     desc = new FieldDescription(resource, name + ".primary-account-number");
55     desc.setRequired(required.contains("primary-account-number"));
56     desc.setRequiredMessage(properties.get(FormsConstants.ELEMENT_PROPERTY_REQUIRED_MSG, title + i18n.get("Credit Card Number is required.")));
57     desc.setConstraintType("foundation/components/form/constraints/luhn");
58     desc.setConstraintMessage(title + i18n.get("The Credit Card Number is invalid. Please check that you introduced the correct Credit Card Number."));
59     FieldHelper.addDescription(slingRequest, desc);
60
61     desc = new FieldDescription(resource, name + ".name-on-card");
62     desc.setRequired(required.contains("name-on-card"));
63     desc.setRequiredMessage(properties.get(FormsConstants.ELEMENT_PROPERTY_REQUIRED_MSG, title + i18n.get("Name on Card is required.")));
64     FieldHelper.addDescription(slingRequest, desc);
65
66     desc = new FieldDescription(resource, name + ".expiration-date-month");
67     desc.setRequired(required.contains("expiration-date"));
68     desc.setRequiredMessage(properties.get(FormsConstants.ELEMENT_PROPERTY_REQUIRED_MSG, title + i18n.get("Expiration Month is required.")));
69     desc.setConstraintType("foundation/components/form/constraints/datenumericmonth");
70     desc.setConstraintMessage(title + i18n.get("The Expiration Month is invalid."));
71     FieldHelper.addDescription(slingRequest, desc);
72
73     desc = new FieldDescription(resource, name + ".expiration-date-year");

```

Below the code editor, the Properties table is displayed for the selected constraint:

Name	Type	Value	Protected	Mandatory	Multiple	Auto Created
1 constraintMessage	String	The Credit Card Number is invalid. Please check that you introduced the correct Credit Card Number.	false	false	false	false
2 hint	String	Validates if the field contains a valid credit card number.	false	false	false	false
3 jcr:created	Date	2014-09-02T15:06:32.436+03:00	true	false	false	true
4 jcr:createdBy	String	admin	true	false	false	true
5 jcr:primaryType	Name	sling:Folder	true	true	false	true
6 jcr:title	String	Luhn Algorithm validator	false	false	false	false
7 sling:resourceType	String	foundation/components/form/constraint	false	false	false	false



Radu Cotescu

@raducotescu



Ending the week with an open-source contribution proposal: ✓. issues.apache.org/jira/browse/SLING-2803
[#sling](#)

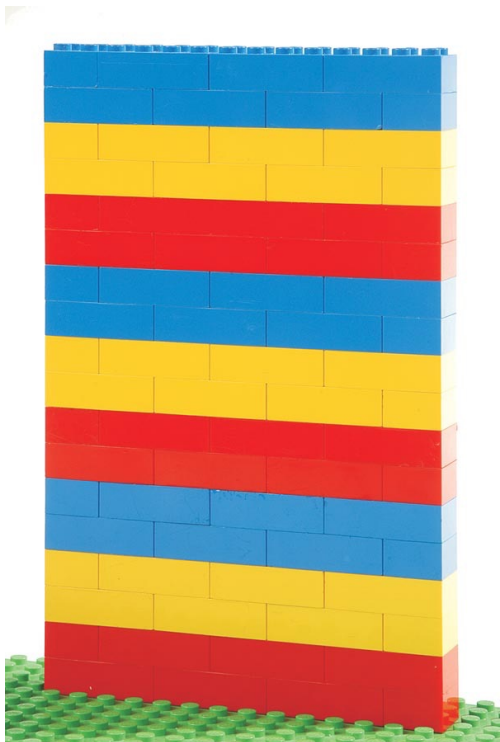
8:13 PM - 28 Jun 2013



The background of the slide is a close-up photograph of various colorful LEGO bricks. The bricks are scattered and overlapping, showing different colors like blue, red, green, yellow, and grey. The lighting is bright, highlighting the texture and details of the plastic bricks.

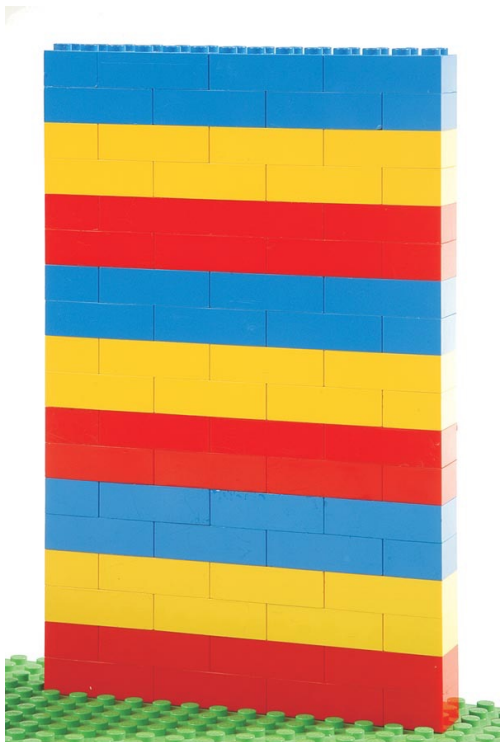
The Building Blocks

The Building Blocks



- ValidationService
- main entry point into the Validation API
 - responsible for retrieving a ValidationModel and for performing the validation operation

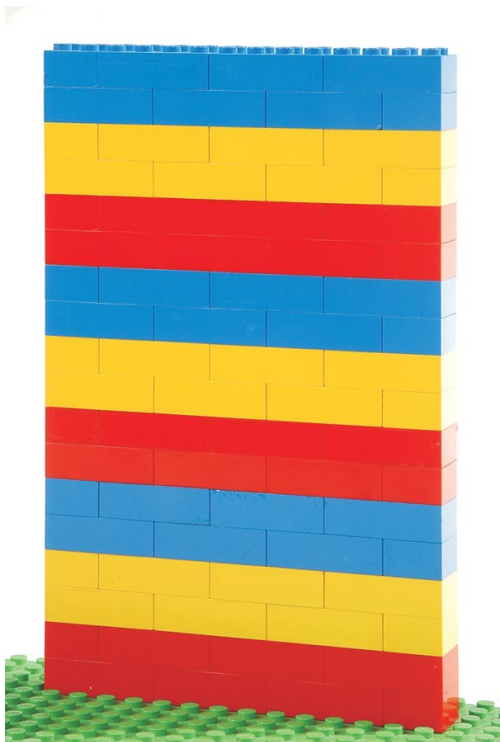
The Building Blocks



ValidationModel

- descriptive structure for the validated object

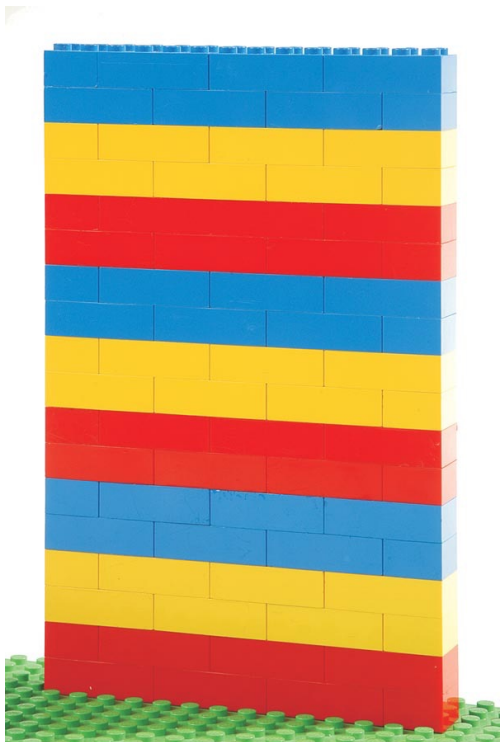
The Building Blocks



ResourceProperty

- describes one of the validated object's properties
- it has a Type and optionally a Validator

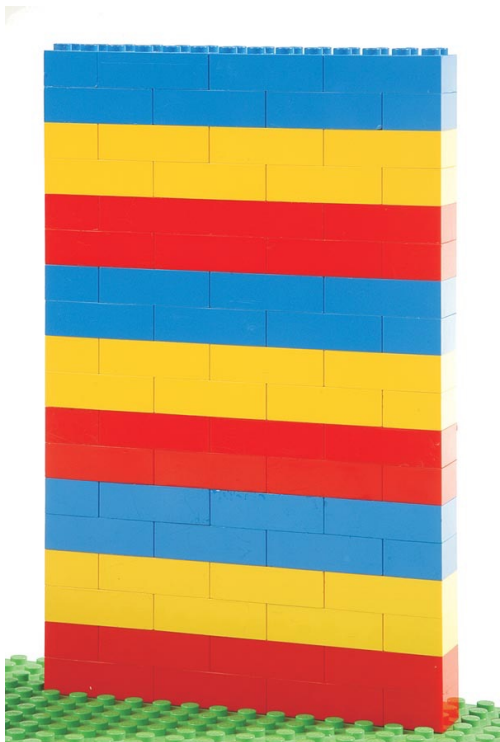
The Building Blocks



Validator

- validates a single piece of information
- can receive arguments

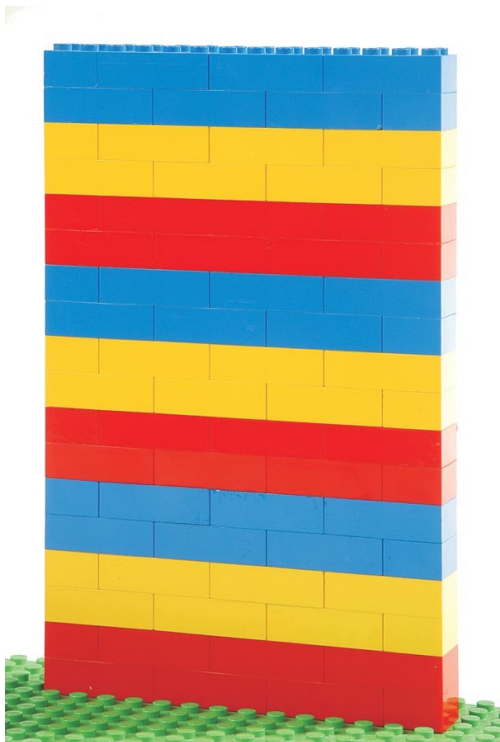
The Building Blocks



ChildResource

- defines validation rules for resource trees
- it's comprised of one or more ResourceProperty objects

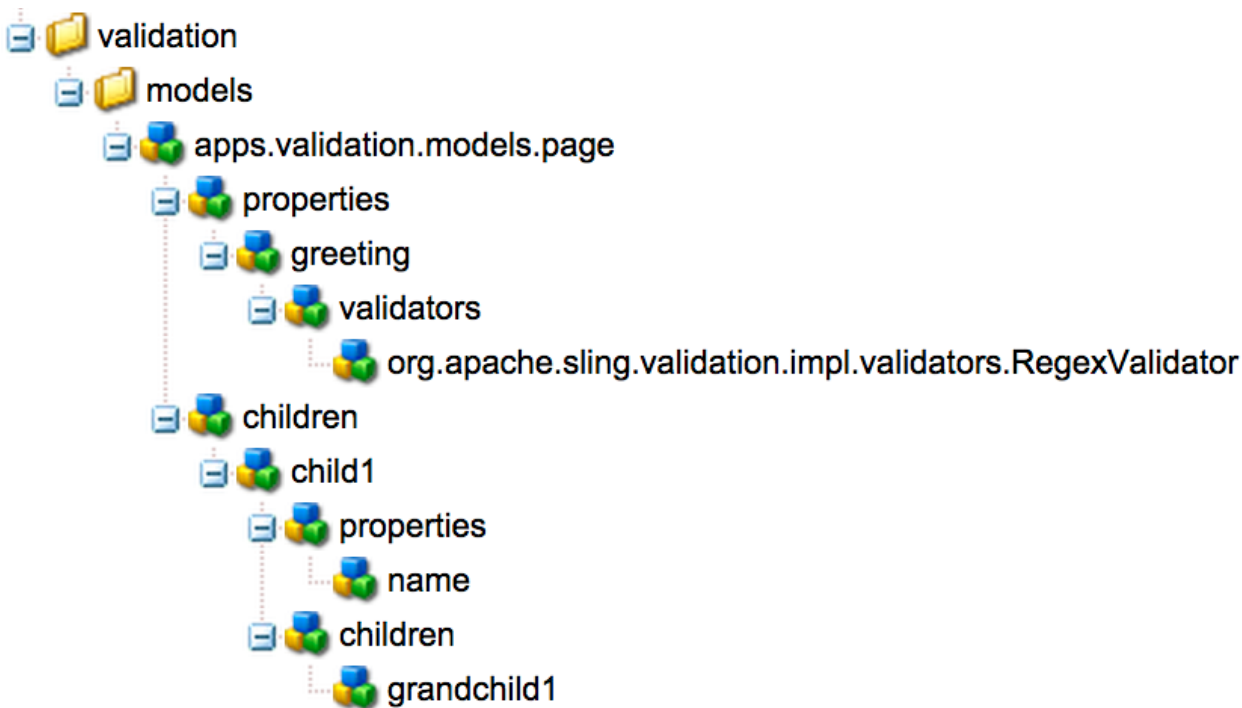
The Building Blocks



ValidationResult

- holds the validation result - boolean
- it can contain validation error messages

Expressing a ValidationModel as content



Expressing a ValidationModel as content

```
apps.validation.model.page
```

```
--applicablePaths=['/content/p/1', '/content/p/2']
```

```
--sling:resourceType=sling/validation/model
```

```
--validatedResourceType=/apps/p/c/page
```

```
greeting
```

```
--propertyType=string
```

```
org.apache.sling.validation.impl.validators.RegexValidator
```

```
--validatorArguments=['regex=^HelloWorld$']
```

```
codeExamples();
```

The ValidationService

```
// resource validation
ValidationModel model = validationService.getValidationModel(resource);
if (model != null) {
    ValidationResult result = validationService.validate(resource, model);
}

// request validation
ValueMap map = request.adaptTo(ValueMap.class);
ValidationModel model =
validationService.getValidationModel(VALIDATED_RESOURCE_TYPE, APPLICABLE_PATH);
if (model != null) {
    ValidationResult result = validationService.validate(map, model);
}
```

Simple integration with Sling Models

```
@PostConstruct
protected void validateResource() {
    ValidationModel vm = validationService.getValidationModel(resource);
    if (vm != null) {
        ValidationResult vr = validationService.validate(resource, vm);
        if (!vr.isValid()) {
            // do your processing here
        }
    }
}
```


Features available today

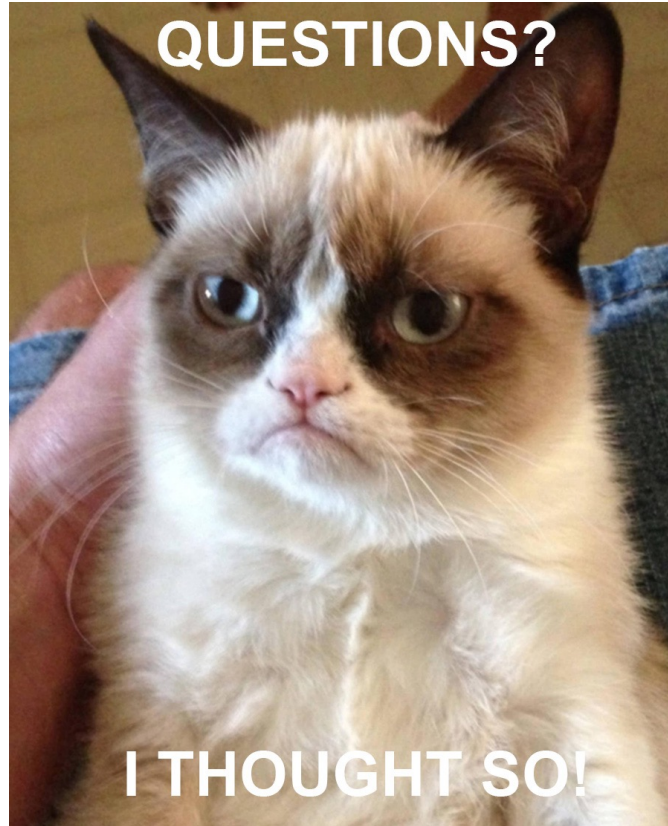
1. ValidationModels based on content structures.
2. The ValidationModels allow resource-tree validation but also request parameters validation.
3. Non-intrusive for existing Sling Models

1. Provide JavaScript validators.
2. Translate `ValidationModel` content structures into JavaScript objects for client-side validation.
3. Define a `Validation` client library.



Demo





Thank you!

Links & Resources

<https://github.com/raducotescu/org.apache.sling.validation>

<https://issues.apache.org/jira/browse/SLING-2803>

[1] https://www.owasp.org/index.php/Data_Validation#Description

The demo artifacts can be found in the `examples` folder from the Git repository.