WebDSL
from a data perspective

Eelco Visser
Software Engineering Research Group
Delft University of Technology
http://eelcovisser.org

RADICAL 2010
Microsoft Research, Cambridge
May 10, 2010
researchr

a data example

http://researchr.org
Researchr

Researchr is a web site for finding, collecting, sharing, and reviewing scientific publications, for researchers by researchers.

Sign up for an account to create a profile with publication list, tag and review your related work, and share bibliographies with your co-authors.
Integration of Data Validation and User Interface Concerns in a DSL for Web Applications


**Abstract**

Data validation rules constitute the constraints that data input and processing must adhere to in addition to the structural constraints imposed by a data model. Web modeling tools do not address data validation concerns explicitly, hampering full code generation and model expressivity. Web application frames works do not offer a consistent interface for data validation. In this paper, we present a solution for the integration of declarative data validation rules with user interface models in the domain of web applications, unifying syntax, mechanisms for error handling, and semantics of validation checks, and covering validation well-formedness, data invariants, input assertions, and action assertions. We have implemented the approach in WebDSL, a domain-specific language for the definition of web applications.

correct & extend
Bibliography: WebDSL

2010

Zef Hemel, Eelco Visser. PIL: A Platform Independent Language for Retargetable DSLs. SLE 2010: [classification]

Danny M. Croonenweghen, Eelco Visser. Integration of Data Validation and User Interface Concerns in a DSL for Web Applications. SLE 2010: 164-173 [doi] [classification]

2009


2008


2007

tagging
reputation system
access control rules
user groups
conference calendar
community engineering
etc.
18,000 lines of WebDSL code

138 (generated) tables in mysql
WebDSL

separation of concerns & linguistic integration
WebLib: a *model* of research
Data Model
data was* the easy part in design of WebDSL

*seemed?
Data Model

entity Publication {
  key :: String (id)
  title :: String (name, searchable)
  authors -> List<Author>
  journal :: String (searchable)
  volume :: Int
  issue :: Int
  year :: Int (searchable)
  month :: String (searchable)
  firstpage :: Int
  lastpage :: Int
  cites -> Set<Publication> (inverse=Publication.cites)
  citedBy -> Set<Publication> (inverse=Publication.cites)
}

entity Author {
  key :: String (id)
  name :: String
  publications -> Set<Publication> (inverse=Publication.authors)
  created :: DateTime
  modified :: DateTime
}
User Interface
Page & Template Definitions

define page root() {
    main{
        section{
            header{"Welcome to WebLib: Your Digital Library"}
            aboutWeblib()
            recent()
            tips()
        }
    }
}

define aboutWeblib() {
    par{"WebLib is a digital library ..."}
}
define no-span main() {  
  <div class="page-wrap">
    top()
    <div class="body">
      messages()
      elements()
    </div>
    <div class="clear"></div>
  </div>
  footer()
}

XML Embedding

Welcome to WebLib: Your Digital Library
WebLib is a digital library ...

Recent Contributions

What Can You Do Here?
- Add an author
- Add a publication
- Search
Page Navigation

define tips() {
  section{
    header{"What Can You Do Here?"}
    list{
      listitem{ navigate(newauthor()){ "Add an author" } }
      listitem{ navigate(newpublication()){ "Add a publication" } }
      listitem{ navigate(search()){ "Search" } }
    }
  }
}

define page search() { ... }
Rendering Data

(queries)
Data Navigation

```javascript
define recent() {
    section{
        header{"Recent Contributions"}
        list{
            for(pub : Publication order by pub.created desc limit 3) {
                listitem{ output(pub) }
            }
        }
    }
}
define output(pub : Publication) {
    citation(pub)
}
```
Rendering Objects

define page publication(pub : Publication) { ... }

define citation(pub : Publication) {
    for(a : Author in pub.authors) { output(a) } separated-by{"", "
    "
    navigate(publication(pub)) { output(pub.title) } ". "
    output(pub.journal) " "
    output(pub.volume) "(" output(pub.issue) "):" 
    output(pub.firstpage) "-" output(pub.lastpage) ","
    output(pub.year)
}
extend entity Person {
  publications -> List<Publication> :=
    select p
    from Publication as p, Author as a
    where (a.person = ~this) and (p = a.publication)
}
Full Text Search

define page search() {
  var query : String

  action show() {
    var pubs : List<Publication> := searchPublication(query);
    if(pubs.length > 0) {
      replace(results, results(query, pubs));
    } else {
      replace(results, noresults());
    }
  }

  main{
    section{
      header{"Search"}
      form{ input(query)[onkeyup:=show()] } 
      placeholder results{}
    }
  }
}
define page publicationsrss(author : Author) {
  rssWrapper {
    <channel>
      <title>"Publications of " output(author.name)</title>
      <link>output(navigate(author(author)))</link>
      <description>"Publications of " output(author.name)</description>
      <pubDate>output(now())</pubDate>
      for(pub : Publication
          in author.publications order by pub.created limit 10) {
        <item>
          <title>output(pub.title)</title>
          <link>output(navigate(publication(pub)))</link>
          <content>citation(pub)</content>
        </item>
      }
    </channel>
  }
}

Changing Data
Forms & Data Binding

define page editpublication(pub : Publication) {
    main{
        section{
            header{"Edit Publication " output(pub.title) ""}
            form{
                group{
                    label("Key:` { input(pub.key) }
                    label("Title:` { input(pub.title) }
                    label("Authors:` { input(pub.authors) }
                }
                submit action{
                    pub.modified := now();
                    return publication(pub);
                } { "Save" }
                }
            }
        }
    }
}
Restricting Access

Access Control Rules

define showReview(review : Review) {
    action delete() { ... }
    section{ ... 
        navigate(editReview(review)){"[Edit]"}
        submitlink delete() {"[X]"}
    }
}
Access Control Rules

```java
define showReview(review : Review) {
    action delete() { ... }
    section{ ... 
        navigate(editReview(review)){"[Edit]"}
        submitlink delete() {"[X]"}
    }
}

access control rules

principal is User with credentials username, password

predicate isReviewer(review : Review) {
    review.reviewer == securityContext.principal
}
rule template showReview(review : Review, pubLink : Bool) {
    review.public || isReviewer(review)
    rule action delete() {
        loggedIn() && isReviewer(review)
    }
}
rule page editReview(review : Review) {
    loggedIn() && isReviewer(review)
}
Data Integrity
Data Validation: Form Input Validation

```javascript
label("Acronym (without year)"){
    input(acronym){

        validate(!isEmptyString(acronym),
            "Provide acronym")

        validate(!(/\s/.find(acronym)),
            "Acronym should not contain spaces")

        validate(!(/[^0-9][0-9][0-9][0-9][0-9]/.find(acronym)),
            "Acronym should not contain year")

    }
}
```
Data Validation: Data Invariants

entity Publication {
    key :: String (id, index(25),

        validate(isUniquePublication(this),
            "that key is already in use"),

        validate(isValidKeyForURL(key),
            "Key should consist of letters, digits, : and -")

    ...  
}

extend entity Person {
    invitation -> Invitation (inverse=Invitation.invitee)

    function invite(email : Email, invitation : WikiText) : Invitation {
        validate(findUser(email) == null
                   && findInvitationByEmail(email).length == 0,
                   "That email address is already in use.");

        ...
    }
}
Implementation
Eclipse IDE built with Spoofax
Compiler

Static analysis

★ check inter-aspect consistency

Compilation by Normalization

★ workflow
★ access control
★ data validation
★ entity extension
★ aspect weaving

Translation to Java
Run-Time System

Data model
- Hibernate
- MySQL
- Lucene

Web server
- Tomcat
- Java servlets
- Lifecycle: decode, bind data, validation, rendering

AJAX
- JavaScript
mapping to Hibernate works

lots of queries!

performance of research is remarkably good

room for improvement: query extraction, prefetching
WebDSL + researchr: platform for RADICAL experiments?

http://webdsl.org