Wiki Widget Editor

SMWCon Spring 2011, Arlington VA, USA
April 29, 2011

Jesse Wang | Ning Hu
What is Widget Editor

Widget Editor is a part of the wiki infrastructure to make Semantic MediaWiki a social semantic web application development platform.

To let people build web applications through metadata and GUI:

- Lowering the bar for people to develop web applications

Still a very work-in-progress prototype (first alpha)
Motivation

Why we want to do this?
Usage of SMW

- Collaboration
- Collecting structured data
- Sharing information
- Management of heterogeneous data
- Simple workflow management
- User-generated arbitrary queries
- Data analysis and information discovery
Common Work Items

- Schema design
- Extension choices
- Forms
- Template
- Skins
- Queries

Expert Indeed
Expert in Need

How can we get more experts?
Vision

It could be a lot easier…
Let’s start with some widgets…
Focus on Content

- Most users just like to build contents
  - CMS: Drupal, WordPress, MW/SMW, …
  - SNS: MySpace, Ning, WetPaint, LinkedIn, …
- Frameworks allows people to focus on content or data
Data and Metadata

- Frameworks let users set “meta-data”
  - So they can customize the content
- Users contribute and customize
  - via meta-data
    - including parameters
- Users can build systems or content repository
  - via meta-data (parameters)
Metadata as Glue

Metadata to glue code and content

It brings power and flexibility in this paradigm:

- More metadata → Higher customizability
- More linkage → Higher flexibility

In Semantic MediaWiki

- Metadata = Special Wiki Data
  - Template, category, property, etc.
Metadata Programming

Existing examples:
- “Allowed values” ➔ Auto-completion in SF
- “has default form” ➔ Map form to data (category)

We want to enhance the experience
- “has range” ➔ auto-completion in Semantic Forms
- “has domain” ➔ auto-construct form for the domain
- “has label/description” ➔ metadata to use in forms
- “has style” ➔ the visual part of customization
Object-Oriented

- OOAD is something quite old now
  - Is OOAD in SMW application development?
- Do we have tools to help user with OOAD?
- Any Design Patterns to help?
- Model-View-Controller (MVVM)
  - Does the model get linked to view nicely?
Category and widget hierarchy
- Each category is a domain (class)
- Each category has a widget
- Widget reflects category hierarchy
  - Animal ➔ Person ➔ Employee, Patient

Widget can embed other widgets
- Aggregation to make application construction easier
- Object “Person” contains object “Address” which has “State”, “City”, etc.
Widget as UI

- Widget has a collection of fields
  - Each field is linked to a property
  - Each field gets metadata from associated property
    - Type (Date, Number, Widget, …)
    - Has domain and range…
- Widget is also a user interface to edit and view wiki content
  - Editing form can be automatically generated
Live Demo

Among the first to see how it works
Demo Recap

- Widget Designer
  - Data type (Number, File/Image, Page, …)
  - Range (number range, enumeration, query)
  - Live widget view
  - Open architecture
  - And more

- Action Connector
  - Action link between widget fields

- Automatic Form Generation
WE Design

How we did it
SMW and OO

- Category ➔ Class
- Property ➔ Attributes
- Page in a category ➔ Instance of a class
- Data in a page ➔ attribute values of an instance

- Bringing OO into SMW is not too hard
SMW and MVC

- Model == category + property + template field?
- View == template?
- Controller == template + parser function?

- More complicated and convolved …
- We are not using *semantic* in SMW
WE View

Display View

Metadata program

Input/Edit View
Field-Internal constraint defined in “Model”
   Property “Allows value”
   “#wfallowsvalue”

Field-External constraint defined by “Action Connector”
   Associate with parser function
The editing view (input form) is automatically generated using the metadata.

User can control:
- Label
- Table or Section (horizontal vs. vertical layout)
- Infobox (with header or not)
- Some styles
- Order (index)
- And more to be added

WE Form
WE Act

- Action Connector to dynamically modify the metadata
- Upon changes in related field’s values
- Provide richer interaction / transaction among the fields (properties)
- All these are meta-data
Discussion

It’s our first demo, we’ll have a lot to do
We need your help.
“What if we build something that nobody uses?” 😊
Will it be useful?
If yes, are you willing to help?
  Help us with development
  Be our first customers
Thank You!

Questions, suggestions and comments?