

# Extreme Forms



William McKee

[william@knowmad.com](mailto:william@knowmad.com)

*WebGUI Users Conference 2009*

# Forms are everywhere.

((( ))) Username  Password

twitter Home Profile Find I

What are you doing? **140**

Latest: working on my WUC presentation. 2 seconds ago



[Advanced Search](#)  
[Preferences](#)  
[Language Tools](#)

Quantity:  4 available

Price: **US \$889.00**

or

Best Offer:

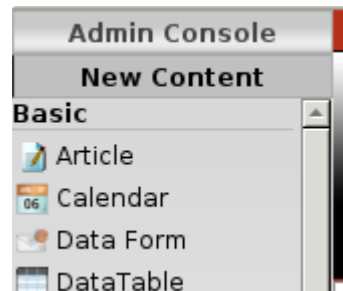
You can also:

Forms make user  
interaction possible.

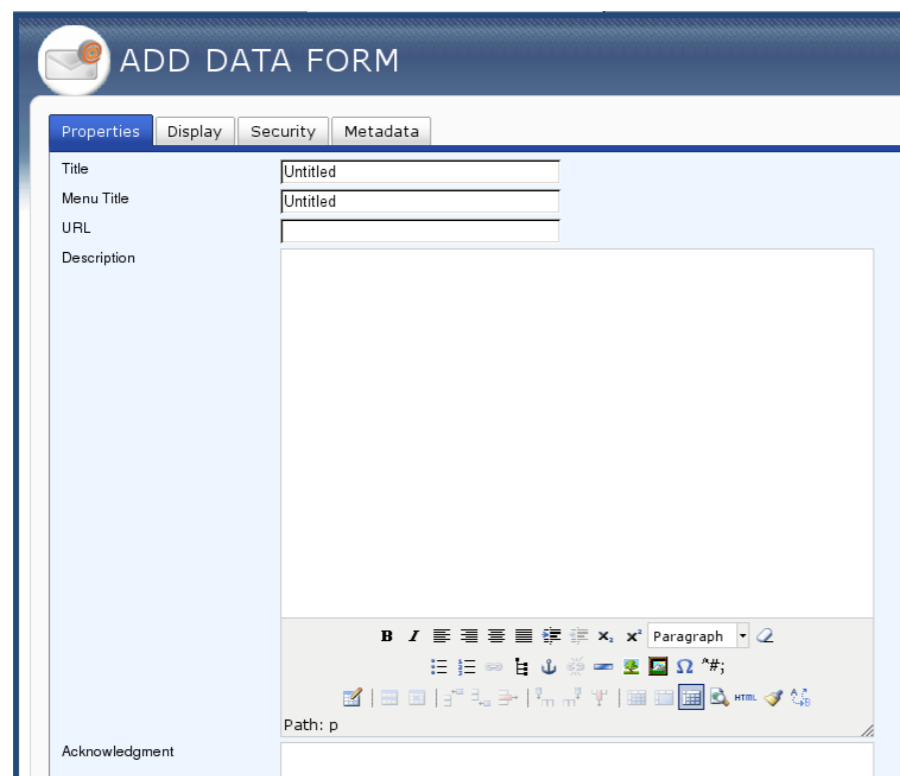
WebGUI makes  
forms easy!

# Building a Form

## 1. Add a Data Form



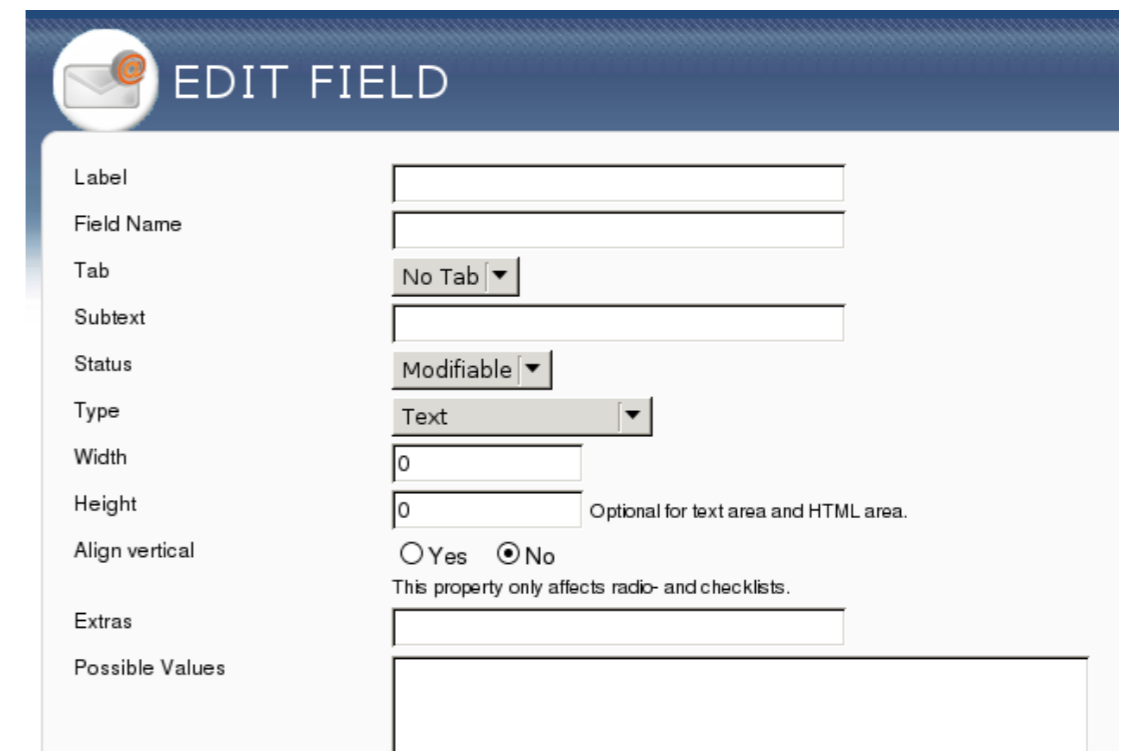
## 2. Define Your Form



## 3. Add a Field

tab delimited. Add a field. Add a Tab

## 4. Define Your Field



## 5. Repeat!

# How can I make **better** forms in WebGUI?

**Start with good planning.**

**Use good markup.**

**Add useful interactive feedback.**

# Plan Your Form



**Start with paper** – draw out the form or use existing paper documents

**Be Brief** – only require necessary fields. Consider your audience.

**Prepare Responses** – error messages, HTML reply, email notification.

# Use good form markup.



**Form Elements** allow the user to enter information.

**Labels** describe fields.

**Fieldsets** group related fields.

**Legends** provide a title for fieldsets.

See Resources for template with proper form markup.



# Interactive Feedback



Feedback  
helps  
your  
users!

# Interactive Feedback

## Character Counter

SAMPLE DATAFORM

Name

Message

256

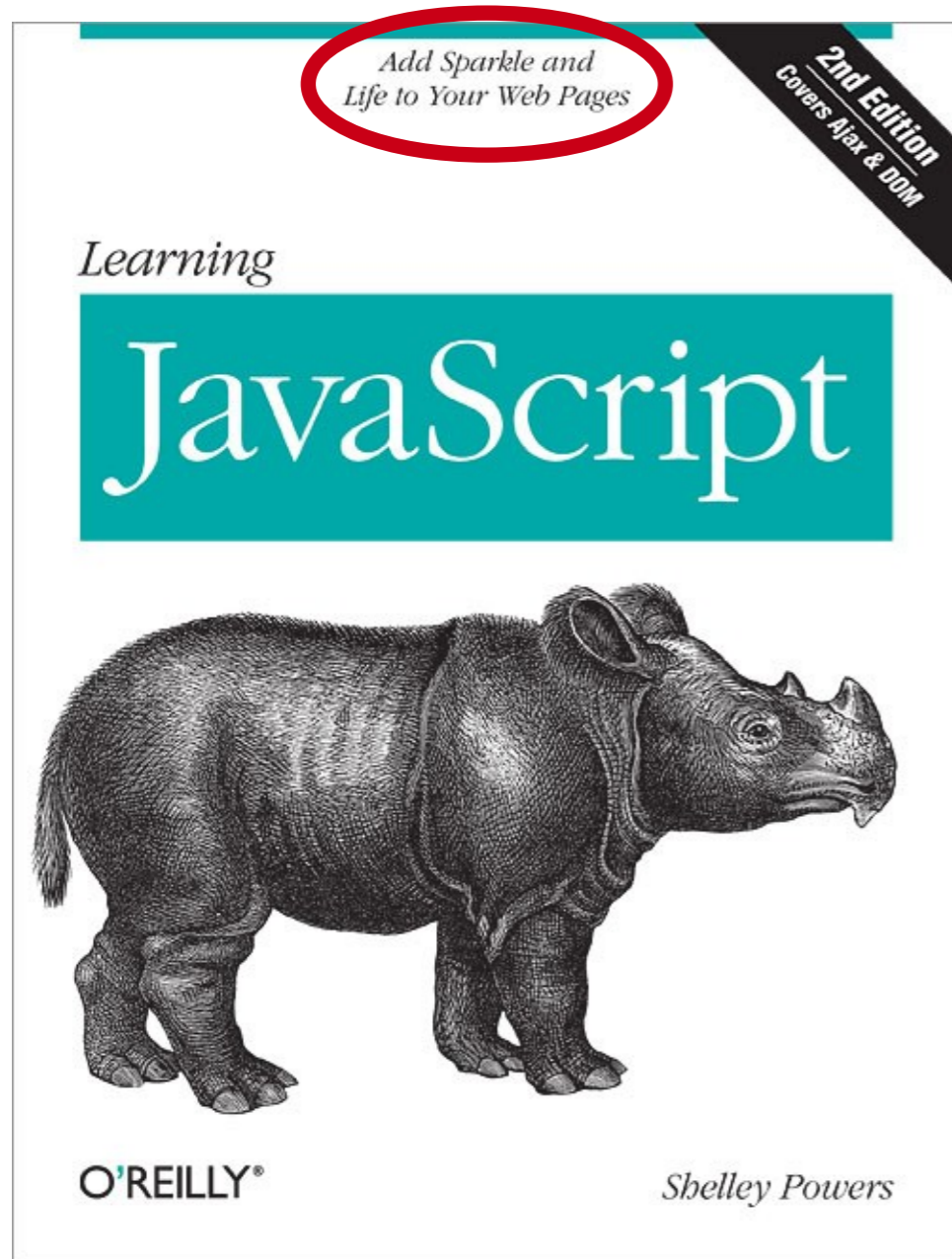
## Inline Error Handling

Email Address \* *Please enter a valid email address.*

## Submit Confirmation

PLEASE WAIT...

# Interactive Feedback



**JavaScript** is your friend!

Each of the previous examples were implemented by adding JavaScript to a custom Form Template.

We use a combination of YUI & JQuery.

# Why stop there?



# How can I make **extreme** forms in WebGUI?

Override tab form behavior

Add dynamic fields

Create custom form controls

# Override Tab Form Behavior

List all entries. \* Export tab delimited. \* Add a field. \* Add a Tab



The screenshot shows a web form titled "ADD NEW TAB" with a blue header bar. On the left side of the form, there are labels for "Label", "Subtext", and "What next?". To the right of "Label" is a single-line text input field. To the right of "Subtext" is a large multi-line text area. Below the "What next?" label is a dropdown menu currently showing "Add new Tab" with a downward arrow. Below the dropdown is a green "save" button.

# Override Tab Form Behavior

## 1. Select the Tab form template

The screenshot shows the 'EDIT DATA FORM' configuration window with the 'Display' tab active. The 'Data Form Template' dropdown menu is open, showing a list of options: 'Tab Form', 'Default Acknowledgement', 'Default DataForm', 'Default Email', 'Mail Form', and 'Data List'. The 'Tab Form' option is highlighted, and a mouse cursor is pointing at it. Other options in the list include 'Style 03', 'Make Page Printable', and 'Data List'. Each option has an 'Edit' and 'Manage' button next to it.

## 2. Default display

The screenshot shows a 'Test Form' with two tabs, 'Tab1' and 'Tab2'. The form contains a 'Name' input field, a 'User' dropdown menu with a 'Select...' option, and a 'save' button.

# Override Tab Form Behavior

## Override Behavior

**Event Details (optional)** ← **Tab (Legend)**

If you want event coverage, please fill in these fields.

<b>Name of Event</b>	<input type="text"/>
<b>Purpose of event</b>	<input type="text"/>
<b>Event Date</b>	<input type="text" value="2009-09-04"/>
<b>Registration instructions</b>	<input type="text"/>

↑ **Labels**

**Order Now**

← **Tab Fields (Fieldset)**

A diagram illustrating the override behavior of a tab form. The form is enclosed in a dashed box. The title 'Event Details (optional)' is highlighted in red, with an arrow pointing to it from a red box labeled 'Tab (Legend)'. Below the title is a subtitle: 'If you want event coverage, please fill in these fields.' The form contains four fields: 'Name of Event', 'Purpose of event', 'Event Date' (with the value '2009-09-04'), and 'Registration instructions'. A red box labeled 'Labels' has an arrow pointing to the labels of the first three fields. A red box labeled 'Tab Fields (Fieldset)' has an arrow pointing to the dashed border of the form. An orange 'Order Now' button is located below the form.



# Build Dynamic Fields

Midwest Heart Location \*

Reason for Appointment \*

Requested Physician

Were you referred by another physician?

If yes, which physician?

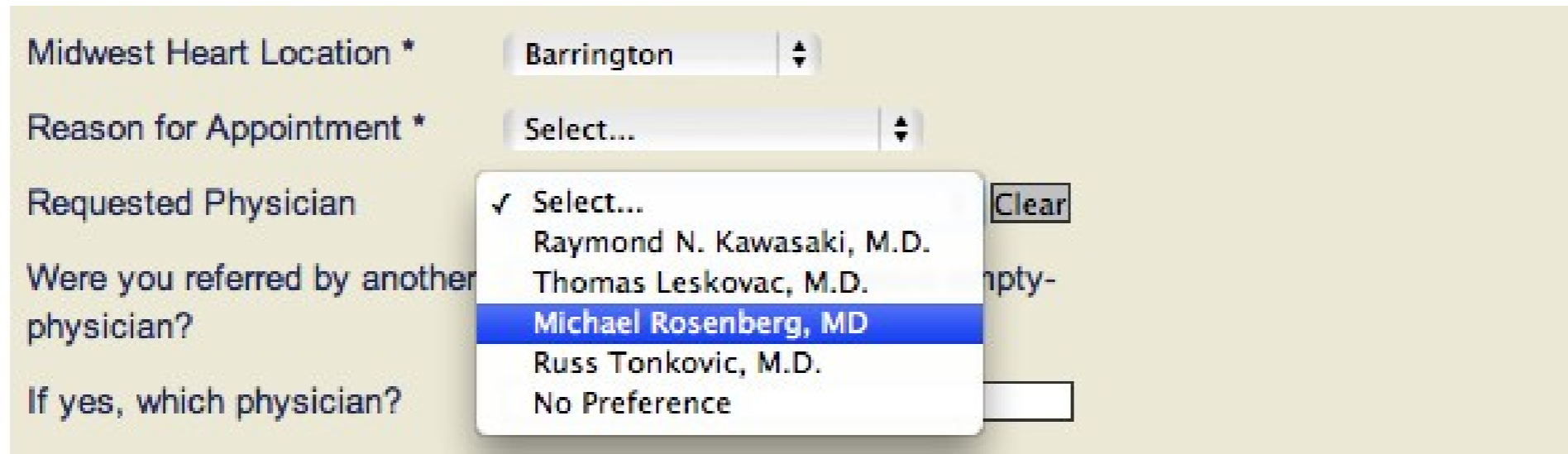
- ✓ Select...
- Raymond N. Kawasaki, M.D.
- Thomas Leskovac, M.D.
- Michael Rosenberg, MD**
- Russ Tonkovic, M.D.
- No Preference

1. Select Location

2. Select a Physician from that location

# Build Dynamic Fields

## Implementation

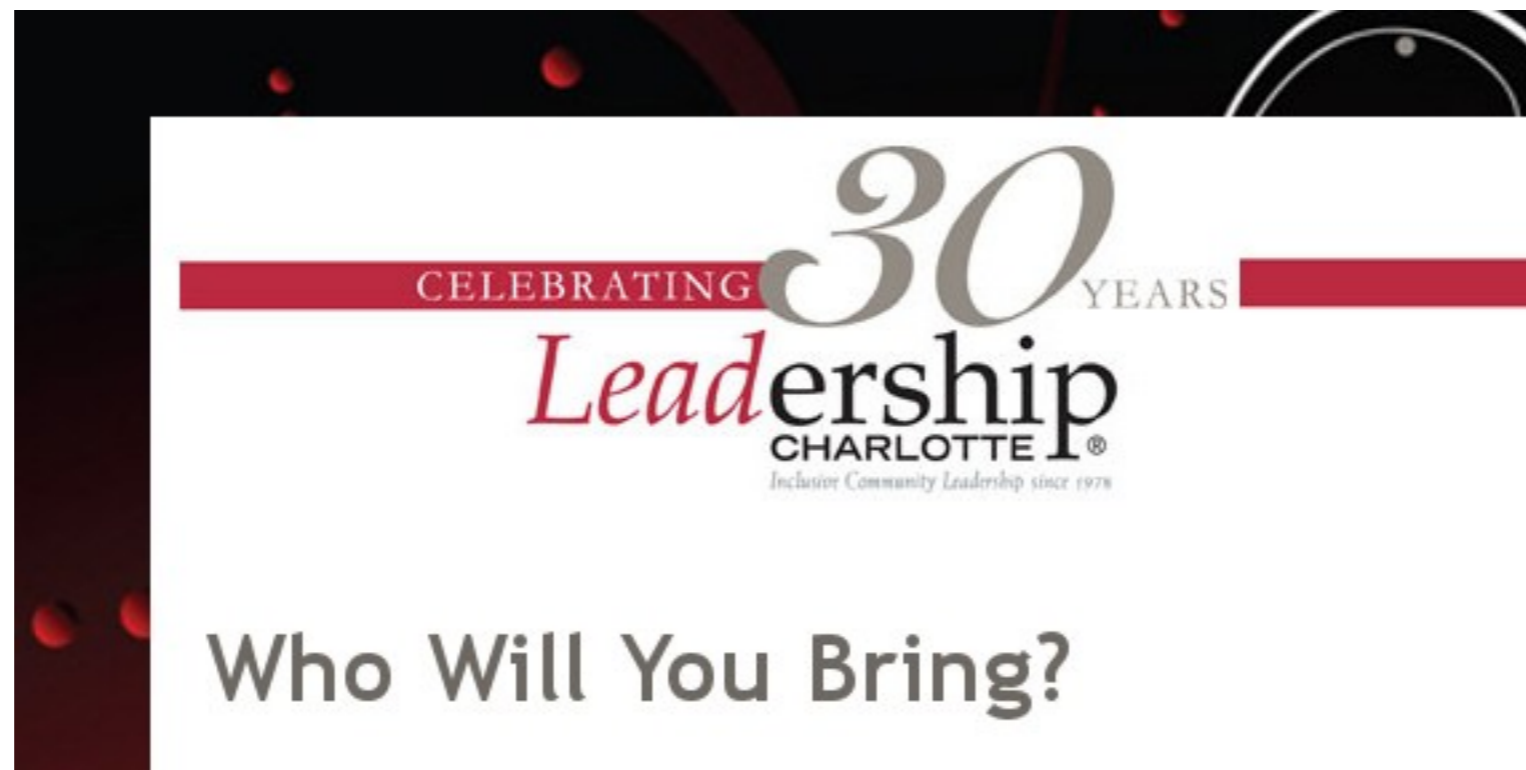


The screenshot shows a web form with several fields. The 'Requested Physician' field is open, displaying a dropdown menu with the following options: 'Select...' (checked), 'Raymond N. Kawasaki, M.D.', 'Thomas Leskovic, M.D.', 'Michael Rosenberg, MD' (highlighted in blue), 'Russ Tonkovic, M.D.', and 'No Preference'. Other fields include 'Midwest Heart Location \*' (Barrington), 'Reason for Appointment \*' (Select...), and 'Were you referred by another physician?' (empty). A 'Clear' button is visible next to the dropdown menu.

1. Use YUI to add an onChange handler to Location
2. Submit a request to an article asset which contains a custom macro with a blank style template
3. Use the response to populate the <select> tag for physician list.

# Custom Form Controls

**Situation:** Our client needed an event management & registration Web site to support their annual volunteer event.



<http://30days.leadershipcharlotte.org>

# Custom Form Controls

**Our Solution:** We identified the following requirements:

- Administrative interface for managing projects.
- List of upcoming events on Homepage.
- Browseable list of all projects.
- Project details page.
- Online registration form.



# Custom Form Controls

**Implementation:** We decided that EMS was not a good fit for the basic registration needs of this project. We built an integrated solution using Thingy, SQL Reports and Dataform.

- Management interface: Thingy
- Upcoming events list: SQL Report
- Browsible project list: SQL Report
- Project details page: SQL Report
- Registration form: Dataform with custom template and custom form control

# Custom Form Controls

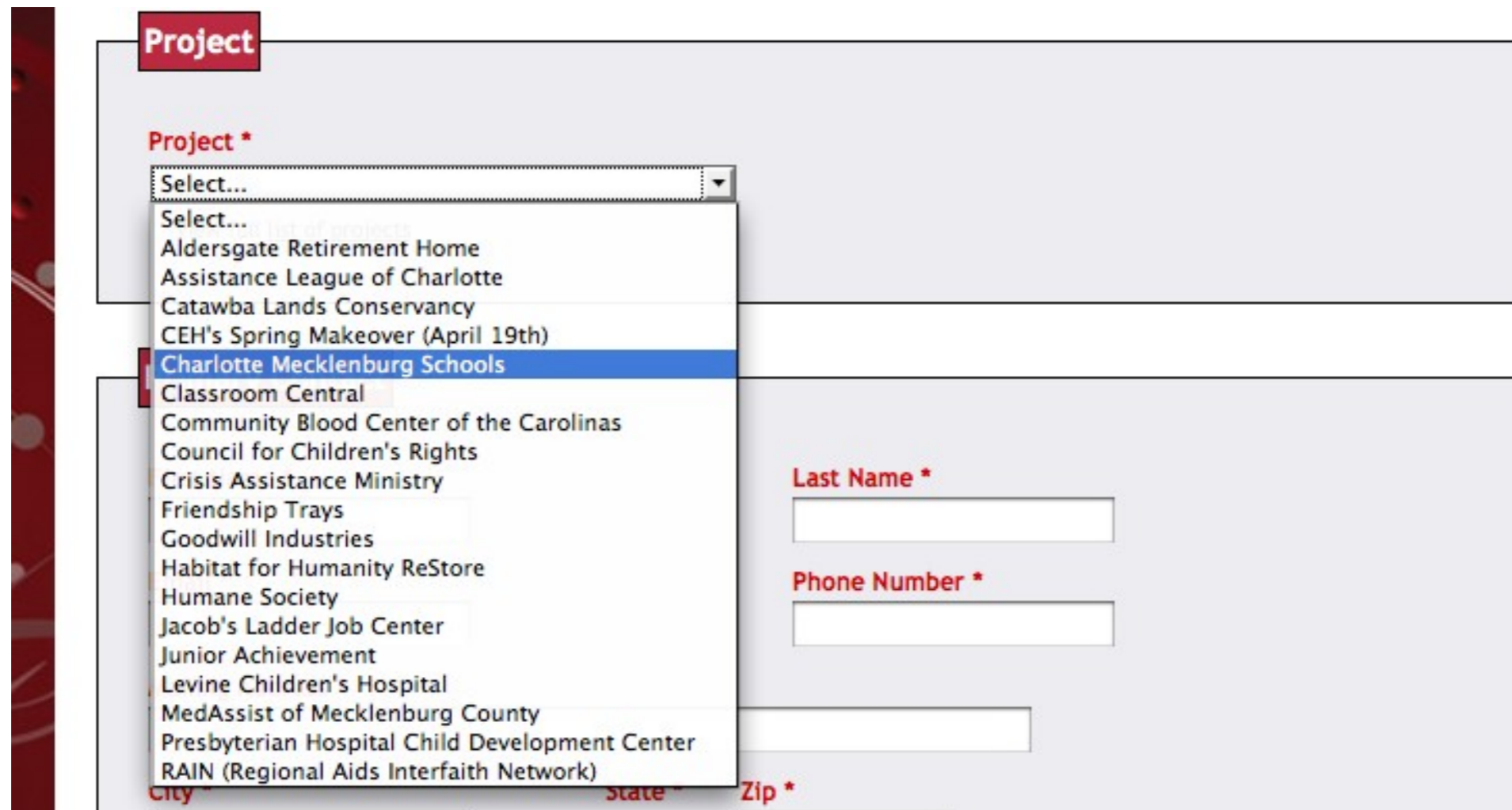
Projects are managed via the Thingy database.

The screenshot shows a web application interface for the 'Events Database'. At the top, there is a dark header with the text 'Events Database • Search Screen Title'. Below this, a navigation bar contains links for 'Manage Things', 'Import', 'Export Projects', and 'Add Projects', followed by a dropdown menu labeled 'Projects'. The main content area is divided into two sections. The first section, titled 'Agency Name', features a text input field and a 'Search' button. The second section, titled 'Search Results', displays a table with the following headers: 'Agency Name', 'Street Address', 'City', 'State', 'Zip', and 'Web Site'.

The Registration form needed a way to select a Project from the list stored in the Thingy database.

# Custom Form Controls

Our custom control creates a select box using data stored in a Thingy table to be displayed in the DataForm asset (whew!)



The image shows a screenshot of a web form. At the top left, there is a red tab labeled "Project". Below it, the label "Project \*" is displayed in red. A dropdown menu is open, showing a list of project names. The first item is "Select..." and the second is "Select...". The third item, "Charlotte Mecklenburg Schools", is highlighted in blue. Other items in the list include "Aldersgate Retirement Home", "Assistance League of Charlotte", "Catawba Lands Conservancy", "CEH's Spring Makeover (April 19th)", "Classroom Central", "Community Blood Center of the Carolinas", "Council for Children's Rights", "Crisis Assistance Ministry", "Friendship Trays", "Goodwill Industries", "Habitat for Humanity ReStore", "Humane Society", "Jacob's Ladder Job Center", "Junior Achievement", "Levine Children's Hospital", "MedAssist of Mecklenburg County", "Presbyterian Hospital Child Development Center", and "RAIN (Regional Aids Interfaith Network)". To the right of the dropdown menu, there are three text input fields. The first is labeled "Last Name \*" in red. The second is labeled "Phone Number \*" in red. The third is labeled "Zip \*" in red. Below the "Zip \*" field, the labels "City" and "state" are partially visible.

# Custom Form Controls

WebGUI has a very modular approach to form controls

All control logic is stored in the WebGUI::Form namespace

These modules are used by Data Form, Thingy, User Profile, etc.



# Custom Form Controls

Gotchas for using Custom Form Controls with the DataForm asset:

- 1) Module must be placed in the core WebGUI library path (e.g., /data/WebGUI/lib/WebGUI/Form)<sup>1</sup>
- 2) The WebGUI::Asset::Wobject::DataForm asset must be modified to include the custom form control.

1. As of WebGUI 7.7.19, this workaround is no longer necessary.

# Other fun ideas:

Use auto-tab in fixed fields.

Pre-populate fields from user profile data.

Store user input with cookies.

# Thank You!



**William McKee**

WebGUI Forums: knowmad

IRC: knowmad

Email: [william@knowmad.com](mailto:william@knowmad.com)

*Resources Available at*  
[www.knowmad.com/wuc2009](http://www.knowmad.com/wuc2009)