

**Forms That Don't Suck**  
(Quick, Easy, & Clean Forms and Data)  
Matt Quackenbush

*cf.Objective() 2011  
Minneapolis, MN  
May 12-14*

## What is this session about?

- ✓ **cfUniForm**
  - a popular, free open source custom tag library for CFML
- ✓ **ValidateThis**
  - a popular, free open source validation framework for CFML

Combining the two to  
“Take the Suck Out of Forms!”

## About Matt Quackenbush

- ✓ Nearly fifteen years CF experience
- ✓ Contributor to various OSS projects
  - cfUniForm (author)
  - ValidateThis
  - Model-Glue
  - ColdBox
- ✓ Blogs (semi-regularly) at [QuackFuzed.com](http://QuackFuzed.com)
- ✓ Freelance consultant (remote/on-site)
- ✓ Band Booster President ([FrenshipBand.com](http://FrenshipBand.com))
- ✓ (Out of practice) Violinist

## Forms Totally Suck

- ✓ Perpetual “Browser Battle”
- ✓ Aggravation of client-side data validation
- ✓ Repetitive writing of server-side data validation

# cfUniForm – Custom Tag Library for CFML

- ✓ CFML wrapper around the Uni-Form markup spec created by Dragan Babic
  - Semantic XHTML compliant markup
  - Consistent display across most (all?) modern day browsers
  - Themed CSS

## **Rich, Consistent Forms with cfUniForm**

- ✓ Write CFML, and don't be concerned with (x)HTML/CSS/JavaScript

# Write CFML, Not HTML/CSS/JS

```
<uform:form action="#cgi.script_name#"
            id="myDemoForm">

    <uform:fieldset legend="Account Registration">
        <uform:field label="Email Address"
                    name="email"
                    isRequired="true"
                    type="text"
                    hint="Note: Please enter a valid email" />

        <uform:field label="Re-enter Email Address"
                    name="email2"
                    isRequired="true"
                    type="text"
                    hint="Note: Please re-type your email." />

        <uform:field label="Choose Password"
                    name="password"
                    isRequired="true"
                    type="password" />

        <uform:field label="Re-enter Password"
                    name="password2"
                    isRequired="true"
                    type="password" />

    </uform:fieldset>
</uform:form>
```

# Write CFML, Not HTML/CSS/JS

## Account Registration

Email Address

\*

Note: Please enter a valid email

Re-enter Email Address

\*

Note: Please re-type your email.

Choose Password

\*

Re-enter Password

\*

Submit



## **Rich, Consistent Forms with cfUniForm**

- ✓ Write CFML, and don't be concerned with (x)HTML/CSS/JavaScript
- ✓ "Inline" or "Block" Layout

# Inline Layout (Default)

<uform:fieldset legend="Account Registration">

## Account Registration

Email Address

\*

Note: Please enter a valid email

Re-enter Email Address

\*

Note: Please re-type your email.

Choose Password

\*

Re-enter Password

\*

Submit

# Block Layout

```
<uform:fieldset legend="Account Registration" class="blockLabels">
```

## Account Registration

\* Email Address

Note: Please enter a valid email

\* Re-enter Email Address

Note: Please re-type your email.

\* Choose Password

\* Re-enter Password

Submit

## **Rich, Consistent Forms with cfUniForm**

- ✓ Write CFML, and don't be concerned with (x)HTML/CSS/JavaScript
- ✓ "Inline" or "Block" Layout
- ✓ Themes that fit into virtually any site layout/scheme.

# Packaged Themes - Default

## Account Registration

Email Address

\*

Note: Please enter a valid email

Re-enter Email Address

\*

Note: Please re-type your email.

Choose Password

\*

Re-enter Password

\*

Submit

# Packaged Themes - Blue

## Account Registration

Email Address

\*

Note: Please enter a valid email

Re-enter Email Address

\*

Note: Please re-type your email.

Choose Password

\*

Re-enter Password

\*

Submit

# Packaged Themes - Dark

## Account Registration

Email Address

\*

Note: Please enter a valid email

Re-enter Email Address

\*

Note: Please re-type your email.

Choose Password

\*

Re-enter Password

\*

Submit

## Rich, Consistent Forms with cfUniForm

- ✓ Write CFML, and don't be concerned with (x)HTML/CSS/JavaScript
- ✓ "Inline" or "Block" Layout
- ✓ Themes that fit into virtually any site layout/scheme.
- ✓ 80% (or more) of the bases covered out-of-the-box.
  - All standard form field types



# Standard Form Fields

Text input

This is a form hint:

---

Textarea

This is a form hint.

---

# Standard Form Fields (cont'd)

Select box

This is a form hint.

Checkboxes

 Option Option Option

This is a form hint.

Radio buttons

 Option Option Option

This is a form hint.

File upload

## Rich, Consistent Forms with cfUniForm

- ✓ Write CFML, and don't be concerned with (x)HTML/CSS/JavaScript
- ✓ "Inline" or "Block" Layout
- ✓ Themes that fit into virtually any site layout/scheme.
- ✓ 80% (or more) of the bases covered out-of-the-box.
  - All standard form field types
  - Utilizes `<cfimage />` to provide CAPTCHA field

# CAPTCHA Field – Powered by <cfimage />

type="captcha"

Please enter the  
letters/numbers you see



## Rich, Consistent Forms with cfUniForm

- ✓ Write CFML, and don't be concerned with (x)HTML/CSS/JavaScript
- ✓ "Inline" or "Block" Layout
- ✓ Themes that fit into virtually any site layout/scheme.
- ✓ 80% (or more) of the bases covered out-of-the-box.
  - All standard form field types
  - Utilizes `<cfimage />` to provide CAPTCHA field
  - Integration with several jQuery plugins to provide "special" field types

# "Special" Fields (jQuery Plugin Integration)

type="date"

\* Date of Birth

Clear		Close				
<Prev	Today	Next>				
March		2011				
Su	Mo	Tu	We	Th	Fr	Sa
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

type="time"

Appointment Time

type="rating"

Rate your picture



A ratings h 4

type="text" mask="(999) 999-9999"


Cell Phone

e.g.: (202) 555-1212

# "Special" Fields (jQuery UI Integration)

type="date"

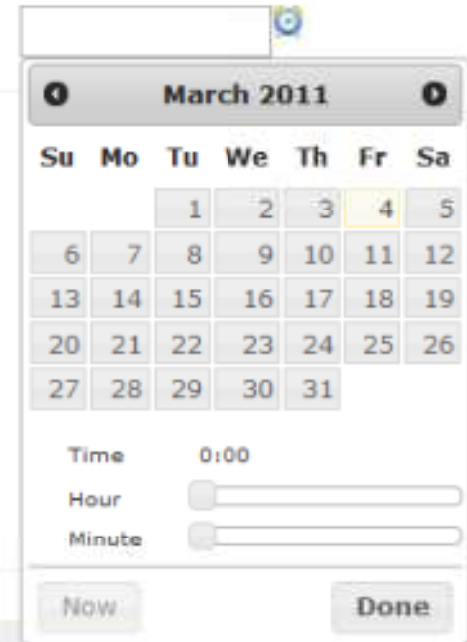
Start Date \*

A calendar widget for March 2011. The days of the week are labeled Su, Mo, Tu, We, Th, Fr, Sa. The date 4 is highlighted in yellow.

type="timestamp"

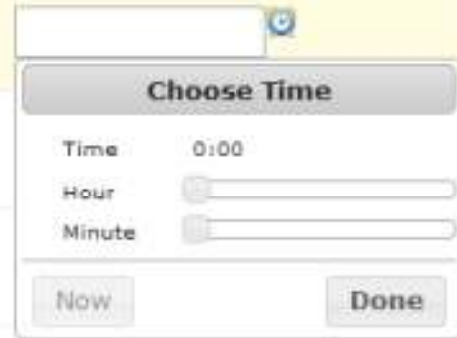
A Timestamp \*

A timestamp widget for March 2011. It includes a calendar grid with the date 4 highlighted, and a time selection section with sliders for Hour and Minute, both set to 0:00. Buttons for 'Now' and 'Done' are at the bottom.

type="time"

Start Time \*

A 'Choose Time' widget. It shows the time 0:00 and sliders for Hour and Minute. Buttons for 'Now' and 'Done' are at the bottom.

## Rich, Consistent Forms with cfUniForm

- ✓ Write CFML, and don't be concerned with (x)HTML/CSS/JavaScript
- ✓ "Inline" or "Block" Layout
- ✓ Themes that fit into virtually any site layout/scheme.
- ✓ 80% (or more) of the bases covered out-of-the-box.
  - All standard form field types
  - Utilizes `<cfimage />` to provide CAPTCHA field
  - Integration with several jQuery plugins to provide "special" field types
- ✓ Customization options to cover most of the other 20%.



# Highly Customizable

Chart Review | Patient Information | Baseline Visit | Intervention Visits | 4-month Followup | 12-month Followup | Contacts

Comments

**INTERVENTION VISITS**

Visit No.	Date	Scheduled Date	Scheduled Time	Method	Duration
1	01/13/2011	<input type="text" value="01/15/2001"/>	<input type="text" value="09:00AM"/>	<input type="text" value="in-person"/>	<input type="text" value="15"/>
2	02/13/2011	<input type="text" value="02/15/2011"/>	<input type="text" value="10:12AM"/>	<input type="text" value="in-person"/>	<input type="text" value="13"/>
3	03/13/2011	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
4	04/13/2011	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

**MAINTENANCE VISITS**

Visit No.	Date	Scheduled Date	Scheduled Time	Method	Duration
1	07/19/2011	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
2	09/19/2011	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
3	11/19/2011	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

|

THANK YOU to Byron Raines for the sample!

## **Clean Data with ValidateThis**

- ✓ 80% (or more) of the bases covered out-of-the-box.

## Out-of-the-Box Validation Support

- ✓ Boolean
- ✓ CollectionSize
- ✓ Custom
- ✓ Date
- ✓ DateRange
- ✓ DoesNotContainOtherProperties
- ✓ Email
- ✓ EqualTo
- ✓ False
- ✓ FutureDate
- ✓ InList
- ✓ Integer
- ✓ IsValidObject
- ✓ Max
- ✓ MaxLength
- ✓ MinPatternsMatch
- ✓ NoHTML
- ✓ NotInList
- ✓ Numeric
- ✓ PastDate
- ✓ Range
- ✓ RangeLength
- ✓ Regex
- ✓ Required
- ✓ Time
- ✓ True
- ✓ URL
- Server-Side Only
  - ✓ Expression

## **Clean Data with ValidateThis**

- ✓ 80% (or more) of the bases covered out-of-the-box.
- ✓ Extremely extensible to cover virtually every use case imaginable.

# Extending ValdateThis ("Custom" Rule)

## ✓ userNameIsAvailable()

```
<property name="UserName">
  <rule type="required" contexts="*" />
  <rule type="rangeLength" contexts="*">
    <param name="minlength" value="3" />
    <param name="maxlength" value="20" />
  </rule>
  <rule type="custom" contexts="New" failureMessage="Sorry, but that User Name is already taken. Please try another one.">
    <param name="methodName" value="userNameIsAvailable" />
    <param name="remoteURL" value="?load=register.checkUName&requestformat=xhr" />
  </rule>
</property>
```

```
/**
 * returns a boolean indication of whether the UserName is available
 */
public boolean function userNameIsAvailable (
)
{
  return getUserService().isUsernameAvailable(userName=getUserName());
}
```

# Extending ValidateThis (Server Rule Validator)

## ✓ ServerNameIsCFObjective

```
component extends="ValidateThis.server.AbstractServerRuleValidator"
{
    /**
     * Verifies that the server name is "CFObjective"
     */
    public string function validate (
        required any valObject
    )
    {
        var serverName = createObject("java","java.net.InetAddress").getLocalHost().getHostName();
        if ( shouldTest(arguments.valObject) && serverName != "CFObjective" )
        {
            fail(arguments.valObject,"You cannot submit this form because the server is improperly named.");
        }
    }
}
```

# Extending ValidateThis (Client Rule Validator)

## ✓ FormIsCFUniFormPowered

```
component extends="ValidateThis.client.jquery.AbstractClientRuleScripter"
{
    public string function generateInitScript (
        string defaultMessage="Only forms rendered by cfUniForm are permitted!",
        string locale=""
    )
    {
        var theScript = "";
        savecontent variable="theScript"
        {
            'function(value,element,options){
                var content = jQuery(".cfUniForm-form-container").contents();
                var rtn = false;
                content.each(function(i,el){
                    if ( this.nodeType == 8 )
                    {
                        var txt = this.nodeValue;
                        if ( txt.search("rendered by cfUniForm") > 0 )
                            rtn = true;
                    }
                });
                return rtn;
            }';
        }
        return generateAddMethod(theScript,arguments.defaultMessage,arguments.locale);
    }
}
```

## Clean Data with ValidateThis

- ✓ 80% (or more) of the bases covered out-of-the-box.
- ✓ Extremely extensible to cover virtually every use case imaginable.
- ✓ Server and Client-side validations from one simple definition file.
  - XML
  - JSON
  - Annotations



# Simple XML Definitions File

```
<?xml version="1.0" encoding="UTF-8"?>
<validateThis xsi:noNamespaceSchemaLocation="validateThis.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <conditions />
  <contexts />
  <objectProperties>
    <property name="FirstName">
      <rule type="required" contexts="*" />
      <rule type="noHTML" contexts="*" />
    </property>
    <property name="LastName">
      <rule type="required" contexts="*" />
      <rule type="noHTML" contexts="*" />
    </property>
    <property name="Email">
      <rule type="required" contexts="*" />
      <rule type="email" contexts="*" />
    </property>
    <property name="Phone">
      <rule type="regex" contexts="*">
        <param name="regex" value="^[\\d]{3}\\.[\\d]{3}\\.[\\d]{4}$" />
      </rule>
    </property>
    <property name="SubjectLine">
      <rule type="required" contexts="*" />
      <rule type="noHTML" contexts="*" />
    </property>
    <property name="MessageBody">
      <rule type="required" contexts="*" />
      <rule type="noHTML" contexts="*" />
    </property>
  </objectProperties>
</validateThis>
```

# **Take the Suck Out of Forms! (Putting it all Together)**

- ✓ Config the Application

# Take the Suck Out of Forms! (Putting it all Together)

- ✓ Application Setup (Without ColdSpring/Bean Factory)

```
public void function onApplicationStart (
    )
{
    var VTConfig = {
        JSRoot="/assets/js/",
        definitionPath="/app/validation/"
    };
    application.ValidateThis = createObject("component","ValidateThis.ValidateThis").init(VTConfig);
    application.cfUniFormConfig = {VT=application.ValidateThis};
}
```

# Take the Suck Out of Forms! (Putting it all Together)

## ✓ Application Setup (With ColdSpring)

```
<bean id="cfUniFormConfig" class="coldspring.beans.factory.config.MapFactoryBean">
  <property name="sourceMap">
    <map>
      <entry key="VT"><ref bean="VT" /></entry>
    </map>
  </property>
</bean>
<bean id="VTConfig" class="coldspring.beans.factory.config.MapFactoryBean">
  <property name="sourceMap">
    <map>
      <entry key="definitionPath">
        <value>/app/validation/</value>
      </entry>
      <entry key="JSRoot">
        <value>/assets/js/</value>
      </entry>
      <entry key="injectResultIntoBO">
        <value>>true</value>
      </entry>
    </map>
  </property>
</bean>
<bean id="VT" class="validatethis.ValidateThis">
  <constructor-arg name="ValidateThisConfig"><ref bean="VTConfig" /></constructor-arg>
</bean>
```

# Take the Suck Out of Forms! (Putting it all Together)

- ✓ Config the Application
- ✓ Config the Form

## Config the Form (Part 1)

```
<cfimport taglib="/kuubd/com/tags/forms/cfUniForm/" prefix="uform" />
<div class="cfUniForm-form-container">
  <uform:form action="contactme.cfm"
    id="my-contact-form"
    errors="#errs#"
    submitValue="Send Message"
    VTtarget="#cf#"
    attributeCollection="#application.cfUniFormConfig#">
```

## Config the Form (Part 2)

```
<p class="bold">Fields marked with an asterisk (*) are required.</p>
<uform:fieldset legend="Contact Info">
  <uform:field label="First Name"
    name="FirstName"
    isRequired="yes"
    type="text"
    value="#cf.getFirstName()"
    hint="" />
  <uform:field label="Last Name"
    name="LastName"
    isRequired="yes"
    type="text"
    value="#cf.getLastName()"
    hint="" />
  <uform:field label="Email Address"
    name="Email"
    isRequired="yes"
    type="text"
    value="#cf.getEmail()"
    hint="" />
  <uform:field label="Phone Number"
    name="Phone"
    isRequired="no"
    type="text"
    value="#cf.getPhone()"
    hint="" />
</uform:fieldset>
```

## Config the Form (Part 3)

```
<uform:fieldset legend="Message Details">
  <uform:field label="Message Topic"
    name="SubjectLine"
    type="select"
    isRequired="true">
    <uform:option display==" Select =="
      value="" />
    <uform:option display="cf.Objective() Awesomesauce"
      value="cf.Objective() Awesomesauce"
      isSelected="#cf.getSubjectLine() IS 'cf.Objective() Awesomesauce'#" />
    <uform:option display="ValidateThis Badassery"
      value="ValidateThis Badassery"
      isSelected="#cf.getSubjectLine() IS 'ValidateThis Badassery'#" />
    <uform:option display="cfUniForm Badassery"
      value="cfUniForm Badassery"
      isSelected="#cf.getSubjectLine() IS 'cfUniForm Badassery'#" />
  </uform:field>
  <uform:field label="Message"
    name="MessageBody"
    isRequired="yes"
    type="textarea"
    value="#cf.getMessageBody()#"
    hint="Type your message here. Please be as detailed as possible. (HTML is not permitted.)" />
</uform:fieldset>
</uform:form>
</div>
```



# Take the Suck Out of Forms! (Putting it all Together)

- ✓ Config the Application
- ✓ Config the Form
- ✓ Config the Controller

# Config the Controller

```
<cfscript>
    result = "";
    cf = createObject("model.ContactForm").init();
    if ( structCount(form) > 0 )
    {
        cf.setFirstName(form.FirstName);
        cf.setLastName(form.LastName);
        cf.setEmail(form.Email);
        cf.setPhone(form.Phone);
        cf.setSubjectLine(form.SubjectLine);
        cf.setMessageBody(form.MessageBody);
        result = application.VT.validate(
                                                    theObject=cf,
                                                    formName="my-demo-form"
                                                    );

        writeDump(form);
    }
    if ( isObject(result) )
    {
        errs = result.getFailures();
    }
    else
    {
        errs = [];
    }
</cfscript>
```

# **Take the Suck Out of Forms!** **(Putting it all Together)**

- ✓ Config the Application
- ✓ Config the Form
- ✓ Config the Controller
- ✓ Enjoy the Benefits of Clean Forms and Data

# Postmortem Summary

The quick and easy path to consistent forms and clean data is simple:

- cfUniForm
- ValidateThis

## Resources

- ✓ RIAForge Downloads  
<http://cfuniform.riaforge.com/>  
<http://validatethis.riaforge.com/>
- ✓ cfUniForm Docs  
<https://www.quackfuzed.com/demos/cfUniForm/>
- ✓ ValidateThis Docs  
<http://www.validatethis.org/docs/>
- ✓ My Blog  
<http://www.quackfuzed.com/>
- ✓ Bob Silverberg's Blog  
<http://www.silverwareconsulting.com/>
- ✓ ValidateThis Google Group  
<http://groups.google.com/group/validatethis/>
- ✓ QuackFuzed@gmail.com