ValidatorWishList

Validator Wish List

A space for wishes validator future functionality.

- **Continuous error-checking with indexed properties** - The validator stops checking for errors when it finds an error with indexed properties. It would be nice if the validator continued checking and caught *all* errors of the indexed property. Maybe even better: make this an optional feature!
- **Multiple-Fields-Validation**- It would be more than cool to apply validation rules

to more than one field (if <field-a.fulfills_condition> and/or <field-b.fulfills_anothercondition>)

- **Documented-Examples** - To become more useful, some documented application examples

apart from those printed in books could be supplied.

- **RegExp field names** - If the property attribute of the <field> element allowed for regular expressions or even simply supported an asterisk for wild card matching.
- **Handling of float, double and bigDecimal** should honor the locale of the user. Converting from a string to a double using <code>new Double(s)</code> will breake outside of USA.

Validator Integration with Frameworks Other Than Struts

I have written an adapter of the Commons-Validator for the Spring project. While considering if the adapter could be released (moved out of the sandbox), it was determined that there were two classes that were outside the scope of Spring that should be maintained as part of the Commons-Validator project. The following suggestions provide an outline of how this can be done without breaking backward compatibility. Both of the classes mentioned below can be found in the Spring sandbox as well as the Struts source code (customized respectively for each framework).

Commons-Validator should maintain the FieldChecks class. Currently FieldChecks is a class that Spring (as well as Struts and any other framework that wants to support Commons-Validator out of the box) must provide to connect Validator to the specific framework's error handling mechanism. It would be nice if the Validator project would abstract the framework-specific part of the FieldChecks class out to a simple interface something like this:

```
public interface ValidationErrorPublisher {
    /**
     * Framework-specific error message publisher.
     *
     * @param field The field to be validated.
     * @param va The ValidatorAction for which the error occurred.
     * @param parameters Parameters passed to the Validator via setParameter().
     */
    void publishError(Field field, ValidatorAction va, Map parameters);
}
```

Then any framework could simply maintain an implementation of this interface instead of the entire FieldChecks class with all of its validation specific code. As a side benefit, a standard version of validation-rules.xml could be provided by Validator because frameworks would no longer have to tell Validator how to call each validation rule.

JavascriptValidatorTag should be maintained with Commons-Validator. Validator should at least provide a version of this class with abstract dependency retrieval methods (getValidatorResources and getMessage) so that a framework could simply extend the abstract class to provide this tag. Ideally, Validator would have its own tag library, but I can't think of how this would be implemented easily.

It would be even better if generation of the validation javascript could be abstracted out into a JavascriptValidatorSupport class that could act as a base for JavascriptValidatorTag and also be usable for integration with other view technologies, like Velocity.

Thanks, Daniel Miller