Upgrade Instructions

Upgrade from 2.x to 3.x

- Instead of using `StructRules` to define struct validation rules, use `ValidateStruct()` to declare and perform struct validation. The following code snippet shows how to modify your code:

```
// 2.x usage
err := validation.StructRules{}.
    Add("Street", validation.Required, validation.Length(5, 50)).
    Add("City", validation.Required, validation.Length(5, 50)).
    Add("State", validation.Required, validation.Match(regexp.MustCompile("^[A-Z]
{2}$"))).
    Add("Zip", validation.Required, validation.Match(regexp.MustCompile("^[0-9]
{5}$"))).
    Validate(a)

// 3.x usage
err := validation.ValidateStruct(&a,
    validation.Field(&a.Street, validation.Required, validation.Length(5, 50)),
    validation.Field(&a.City, validation.Required, validation.Length(5, 50)),
    validation.Field(&a.State, validation.Required,
validation.Match(regexp.MustCompile("^[A-Z]{2}$"))),
    validation.Field(&a.Zip, validation.Required,
validation.Match(regexp.MustCompile("^[0-9]{5}$"))),
)
```

- Instead of using `Rules` to declare a rule list and use it to validate a value, call `Validate()` with the rules directly.

```
data := "example"

// 2.x usage
rules := validation.Rules{
    validation.Required,
    validation.Length(5, 100),
    is.URL,
}
err := rules.Validate(data)

// 3.x usage
err := validation.Validate(data,
    validation.Required,
    validation.Length(5, 100),
    is.URL,
)
```

- The default struct tags used for determining error keys is changed from `validation` to `json`. You may modify `validation.ErrorTag` to change it back.