

Module 6 sample

Validating data with Validator

6.1 The data validation process.....	1
6.2 The Validator component	3
6.3 Validator in action	6
6.4 Summary	25
Index	26

In the world of programming, once in a while a nifty idea comes along—a valuable tool or a spark of imagination that makes you wonder how you ever programmed without it. Commons Validator is one such tool; once you start using it, it’s difficult to program without it.

Validator provides a guideline and API for validating user data. It forces the developer to centralize and streamline data validation and lays down the best possible way to manage this validation. At the same time, it provides an API that brings order to data validation. It really is the tool that you can’t live without, if your application regularly accepts user data.

In this module, we’ll start with a little background about data validation—why you should do it, how to do it, and what should be done. We’ll then move on to discuss the Validator component, giving an overview and discussing its API. This will prepare you for the Validator examples that follow. You’ll see how to use Validator in several situations, including web-based and normal applications. In the web applications, we’ll consider Struts and non-Struts applications, to cover Validator usage in almost all situations you’re likely to encounter.

6.1 The data validation process

Computers are dumb. Yes, that’s right. Computers are dumb machines that do what you and I tell them to do. So why is it so difficult for us to make them work the way we want them to? Is the problem something we do, or something we don’t do? The answer isn’t easy to figure out. How many of us have spent hours of frustrating detective work trying to figure out what the computer is doing, only to realize that it was after all, a stupid mistake on our part that made it behave that way? Yes, it’s not computers that are dumb: It’s us.

With that frightening realization out of the way, let’s see how we can minimize the time and effort required to figure out problems in our code. Since every application depends on the data it receives, the first salvo to fire is targeted toward making sure this data is validated to conform to set guidelines. This is why data validation processes are the most important facets of any application.

Let’s start our discussion of the process with an understanding of why data validation should be performed.

6.1.2 Why bother?

In simple terms, data should be validated because users are unpredictable. You can’t rely on users to consistently provide you with valid data that’s applicable for your application. Even the most fastidious users will tire and input invalid numbers in a salary field, for example. However, data should be validated for a number of reasons to cover a variety of issues that you’re likely to encounter:

- To cater to unpredictable users. For example, a user may enter a `String` value in a `Number`-only field, due either to ignorance or a genuine mistake.
- To constantly protect your application from rogue and mischievous users.
- To ensure the integrity of data already in your system. There is nothing more frustrating than sifting through your application's data store for invalid data that has the potential to render your data systems invalid.
- To ensure conformity and compliance with existing systems.
- To reduce application-processing time. In most systems, frameworks exist to separate the task of business processing and front-end interactions into separate modules. If data validation at the front end can sift through the user input and invalidate entries that don't conform to set guidelines, it reduces the processing that business units have to do.

This is by no means an exhaustive list of the reasons for data validation, but it covers the most prominent ones. In the next section, we'll discuss how to perform data validation.

6.1.2 The how-to of data validation

We all realize the need to perform data validation and agree that's a required part of the application development process. But how do we perform validation? Following are some basic guidelines for this process:

- The first step in data validation should be performed at the data-entry point. There is no sense in holding data and performing validation at a later stage, because any subsequent processing can be avoided if the supplied data fails. The only exception to this rule arises when the validity of data depends on subsequent data collection or business process.
- You should take into account existing criteria for data-set validation. This means that validation can only be performed against rules governing a collection of previous data. These rules must be explicit and precise and should be made available to the validation routines before data is gathered from the user.
- The data validation rules must also be separated from the source code so you can modify the validation rules without affecting applications that are in production mode. Therefore, these rules must be supplied to the validation routines as an external process that's only looked up at the precise moment the validation is to be performed.
- The data validation process should also make sure the user or system supplying the data can be made aware if any data validation checks and consequently can be allowed to resubmit with correct data or gracefully withdraw from continuing with the application.
- The data validation routines must be independent of the surrounding process. This means that the same validation routines should work identically in all applications, regardless of the way input is supplied to the routines.

Let's now discuss what to look for when you're validating data.