

# Validating Form Data

---

## Make form fields required

1. Open **ReservationForm.mxml** and locate the **Form** container.
2. Locate the first **FormItem** tag.
3. Add a `required` property with a value of `true`.

```
<mx:FormItem label="Full Name:" required="true">
    <mx:TextInput id="fullname"/>
</mx:FormItem>
```

4. Save the file and run.  
You should see a red asterisk next to the Full Name field indicating that it is required.
5. Make the **Address**, **City**, **State**, **Postal Code**, **Phone**, and **Date Needed** fields required also.

## Create validation

6. Locate the **Button** control and add an `id` property of button.  

```
<mx:Button label="Submit" id="button"/>
```
7. After the **Script** block, create a `StringValidator` tag and use the following properties to validate the **fullname** text input.

- `id` with a value of `fullnameV`
- `source` bound to `fullname`
- `property` with a value of `text`
- `trigger` bound to `button`

Your code should appear as follows:

```
<mx:StringValidator id="fullnameV"
    source="{fullname}" property="text"
    trigger="{button}"/>
```

8. After the **StringValidator**, add another StringValidator with an id of addressV. Add the following properties:
  - source bound to address
  - property with a value of text
  - trigger bound to button
9. Add another StringValidator with an id of cityV. Add the following properties:
  - source bound to city
  - property with a value of text
  - trigger bound to button
10. Add another StringValidator with an id of stateV. Add the following properties:
  - source bound to state
  - property with a value of text
  - trigger bound to button
11. Add another ZipCodeValidator with an id of postalcodeV. Add the following properties:
  - source bound to postalcode
  - property with a value of text
  - trigger bound to button
12. Add a PhoneNumberValidator with an id of phoneV. Add the following properties:
  - source bound to phone
  - property with a value of text
  - allowedFormatChars with a value of "-" (dash)
  - trigger bound to button
13. Add a DateValidator with an id of dateNeededV. Add the following properties:
  - source bound to dateNeeded
  - property with a value of text

- `allowedFormatChars` with a value of `"/"` (forward slash)
- `trigger` **bound to** button

Your code should look like this:

```
<mx:StringValidator id="fullnameV" source="{fullname}"
    property="text" trigger="{button}" />
<mx:StringValidator id="addressV" source="{address}"
    property="text" trigger="{button}" />
<mx:StringValidator id="cityV" source="{city}"
    property="text" trigger="{button}" />
<mx:StringValidator id="stateV" source="{state}"
    property="text" trigger="{button}" />
<mx:ZipCodeValidator id="postalcodeV"
    source="{postalcode}" property="text"
    trigger="{button}" />
<mx:PhoneNumberValidator id="phoneV" source="{phone}"
    property="text" allowedFormatChars="-"
    trigger="{button}" />
<mx:DateValidator id="dateNeededV"
    source="{dateNeeded}" property="text"
    allowedFormatChars="/" trigger="{button}"/>
```

14. Locate the **Script** block.
15. After the last import statement, import the `ValidationResultEvent` class.
16. Also import the `Validator` class.
 

```
import mx.events.ValidationResultEvent;
import mx.validators.Validator;
```
17. Create and instantiate a private variable named `myValidators` datatyped as `Array`.

```
private var myValidators:Array = new Array();
```

18. Locate the **init()** function and after the event listener code, list using bracket notation all the validator instance names and assign to myValidators.

```
myValidators = [fullnameV, addressV, cityV, stateV, postalcodeV, phoneV, dateNeededV];
```

19. Before the end of the **Script** block, create a private function named validateForm that takes no parameters and returns void.

20. Within the function, create and instantiate a local variable named vals datatyped as Array.

```
var vals:Array = new Array();
```

21. Invoke the validateAll() method of the Validator class and pass myValidators as the only parameter. Assign it to vals.

```
private function validateForm():void{  
    var vals:Array = new Array();  
    vals = Validator.validateAll(myValidators);  
}
```

22. Locate the **Button** control within the **Form** container.

23. Add a click event named validateForm that does not pass any parameters.

```
<mx:Button id="button"  
    label="Submit"  
    click="validateForm()"/>
```

24. Save the file.

25. Return to **AdobeODT.mxml** and run.

Click the button in the form. All the fields display a red border.

The image shows a web form with three main sections:

- Contact Information:** Contains six text input fields, each with a red asterisk indicating a required field. The fields are labeled: Full Name, Address, City, State, Postal Code, and Phone.
- Room Information:** Contains one date input field labeled "Date Needed" with a red asterisk and a small calendar icon to its right.
- Selected Options:** A section with the label "Selected Options:" and a large empty area below it.

At the bottom center of the form is a "Submit" button.

## Restrict date selection

26. Locate the **DateField** in the **Form**.
27. Add a `selectableRange` property bound to the value of the **Object** `{rangeStart : new Date()}`

```
<mx:DateField id="dateNeeded"
    selectableRange="{rangeStart : new Date()}" />
```

28. Save the file and run.

Click the date field icon. You should see all days before the current date disabled.