# Data Formatting of Fields in DSPs

Blog by Barbara Douma and Jyoti Singh

Dynamic server pages (DSPs) provide the user interface for web applications. They enable users to view and modify data in a browser, and provide a lot of built-in functionality to ensure that data is formatted and displayed correctly, and to ensure that data entered by the user is correct before it is stored.

For example, a DSP automatically checks that a Numeric field does not contain alphabetic characters. This happens in the browser, in the client side of the DSP, even before the data is submitted to the backend server, where the data is validated again before being stored. Having the client side of the DSP handle formatting and syntax checking prevents unnecessary round trips to the server.

DSPs provide a lot of default data formatting out the box, to ensure that the data is correctly displayed. For example, a Numeric field with four decimals may be formatted to show only two decimals.

By default, for fields with an EditBox widget (or any widget mapped to and HTML input of type text):

- Uniface non-String fields (Numeric,  Float, Date, Time, Datetime, Boolean) are formatted on the DSP client in the browser. The client applies supported display formats as defined by the field's Field Layout property.
- Uniface String fields are formatted in the server-side of the DSP by the Uniface Server. The server applies any display formats defined by the field's **Field Layout** property before sending the formatted String to the browser.

(For more information on DSP widgets, see the widget reference documentation in the *Uniface Library*.)

In most cases, its is best to rely on default behavior for formatting non-String fields in DSPs. By sending unformatted data to the server, the original data remains available in the DSP client's data layer, and the client has all the information it needs to format the data for display, but also to deformat it for transport back to server.

However, there can be circumstances when you want to use server-side formatting. For example:

- You might want to specify a Uniface display format that is not supported by client-side formatting in DSPs, such as a minus sign for a negative number.
- You might want to customize behaviour, such as implementing a custom date/datetime picker using JavaScript instead of relying on browser feature support.
- The data is read-only and shouldn't be formatted in the browser.

## Client-Side Formatting

In client-side formatting, the Uniface Server sends the field's value to the client as unformatted data, and expects to get unformatted data back.

The DSP client formats the received data using its display format (**Field Layout**), and assigns the formatted values to the widget for display. If the value of a field is changed in the browser (either by the user editing the field or some code that assigns a new value), the DSP client:

- Deformats the changed value using the display format definition (in the **Field Layout** property).
- Validates the field syntax using the deformatted value (assuming the widget property **Syntax Check on Browser** is enabled).
- If syntax validation fails, it displays or returns an error, and prevents the data from being submitted to the server until it is corrected.
- If syntax validation succeeds, the new value is formatted again to get displayed, and the data can be submitted to the server.

Once the data is submitted, the data is validated on the Uniface Server, either through an explicit validate statement or when stored to the database.

Client-side formatting and deformatting allows the user to get immediate feedback on any changes—the DSP client updates the displayed value and reports errors. This prevents unnecessary round trips to the server to get the same feedback.

The disadvantage is that gives the developer less control.

## Server-Side Formatting

In server-side formatting, the Uniface Server formats the data before sending it to the DSP client, and deformats the data when it is received from the client so that it can be stored

As mentioned before, String fields are always formatted on the server rather than the client, but it is now possible to do this for non-String fields too. The advantage is that it gives the developer more control over formatting, but it also means the developer is responsible ensuring that data is correctly deformatted and validated.

To do so, you can use the new widget property **Data Formatting on Server** (serverDataFormatting).

> ***Note****: This is the new name for a previously undocumented property: datatype=string. This property continues to work for compatibility purposes.*

When a non-String field with **Data Formatting on Server** set to true, the Uniface Server:

- Formats the field value using the display format (defined in **Field Layout** property).
- Executes trigger formatToDisplay, if defined. This trigger enables the developer to complete customize the data formatting.
- Sends the resulting formatted string to the DSP client

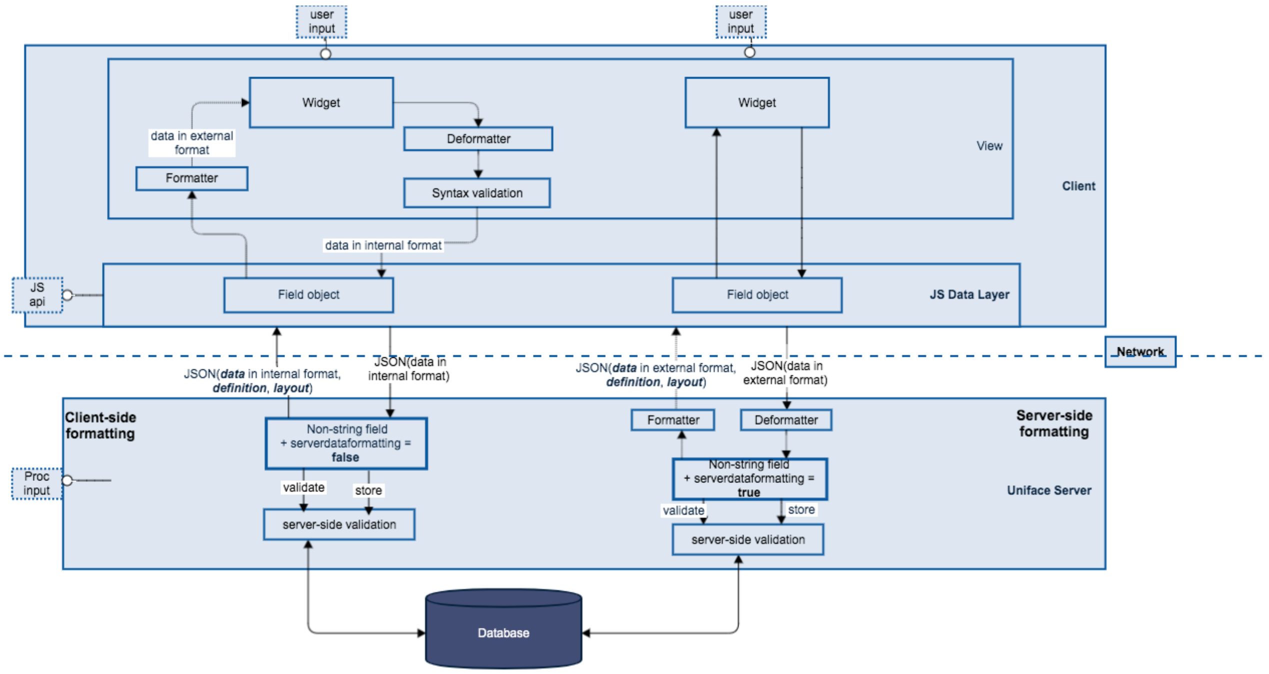
The DSP client assigns the formatted string to the widget for display. If the field value changes on the DSP client, the client ***does not*** deformat or validate the data (the widget property **Syntax Check on Browser** is ignored).  This is because the DSP client does not have the information needed for deformatting, such as the NLS locale and display formats used.

When the DSP client submits its data for which server-side formatting is enabled, it sends the field data to the Uniface Server as is. The Uniface Server then:

- Deformats the field data using the **Field Layout** definition.
- Executes trigger formatFromDisplay, if defined. This trigger enables the developer to control how the data is deformatted.
- Validates the deformatted value, either in response to an explicit validate statement or when stored to the database.
- Returns a response to the DSP client with the updated data or validation errors.

Server-side formatting enables the developer to have complete control over formatting and deformatting but it means that a full round-trip to the server is required before the display can be updated or errors reported.

The following diagram shows the data formatting mechanisms in DSP:



## Property: Data Formatting on Server (serverdataformatting)

Server-side formatting is made possible by the widget property **Data Formatting on Server**. This is the new name for a previously undocumented property: datatype=string. This property continues to work for compatibility purposes.

Other characteristics:

- Default value: false (no server-side data formatting)
- Not applicable to fields with Uniface data type String. String fields are always formatted/deformatted on the server
- If set to true, the widget property **Syntax Check on Browser** is ignored

For more information about this property, see the Uniface documentation.

## Start Using the New Property

The configuration file for the new property will be delivered with the next Uniface release and not with a patch. So to have the new property displayed in the DSP Widget Properties dialog, you need to install the patch and then manually edit the file **uproperties.ini**:

- Open the file **uproperties.ini**, located in ***UnifaceInstallDir*\uniface\adm**
- In the section [propertygroups], add the property serverdataformatting to grp:uniface_all .
- Add the following property definition at the end of the file:

```
[serverdataformatting]
description=Data Formatting on Server
datatype=boolean
tooltip=Format the field data on the server in string type
category=Uniface
```