

Providing Converters and Validators from Plugins

Column converters as well as validators can be provided via client plugins. See [Creating Client Plugins](#) for more information on how to create client plugins.

Column Converters

In order to build a column converter, the plugin class implementing the `IClientPlugin` interface (see [Implement the Plugin Interface](#)) also needs to implement the interface `IColumnConverterProvider` to produce `ITypedColumnConverter` instances in the `getColumnConverters()` method. See [API docs](#).



Note that `IColumnConverter` is considered to be deprecated, therefore in `getColumnConverters()` method do return an `ITypedColumnConverter` array.

A column converter class implementing the `ITypedColumnConverter` interface has to implement the following three methods that will be displayed in the **Conversion** tab:

- `convertFromObject()`
Converts from dataprovider value to db value
- `convertToObject()`
Converts from db value to dataprovider value
- `getToObjectType()`
The dataprovider data type (so the resulting type of the `convertToObject()`), being one of the [IColumnTypes](#) constants: `TEXT`, `INTEGER`, `NUMBER`, `DATETIME` or `MEDIA`

A column converter class can define properties in the `getDefaultProperties()` method, which will be displayed in the **Conversion** tab as well.

The column types that the column converter supports are to be specified in the `getSupportedColumnTypes()` method.

For an example of a column converter, see the `NrToJodaConverter` class in the example given on [Providing UI Converters from Plugins](#) page.

All the implemented custom column converters must be returned by the `getColumnConverters()` method of the plugin class implementing the `IColumnConverterProvider` interface, so that they become available within Servoy Developer.

Column Validators

In order to build a column validator, the plugin class implementing the `IClientPlugin` interface (see [Implement the Plugin Interface](#)) also needs to implement the interface `IColumnValidatorProvider` to produce `IColumnValidator` instances in the `getColumnValidators()` method. See [API docs](#).

The validation rule of an `IColumnValidator` is defined in the `validate()` method.

An `IColumnValidator` can also define properties in the `getDefaultProperties()` method, whose values will be set by the developer in UI, in the **Validation** tab.

The column types that the validator supports are specified in the `getSupportedColumnTypes()` method.

All the implemented custom column validators must be returned by the `getColumnValidators()` method of the plugin class implementing the `IColumnValidatorProvider` interface, so that they become available within Servoy Developer.

Example

This is an example of validating the size of a column of type `TEXT` or `MEDIA`. The validation rule is that the size must be lower or equal to the value given by the developer by setting the `length` property in the **Validation** tab.

```
public class SizeValidator implements IColumnValidator
{
    private static final String
        LENGTH_PROPERTY = "length";

    public Map getDefaultProperties()
    {
        Map props = new HashMap();
        props.put(LENGTH_PROPERTY, "");
        return props;
    }

    public String getName()
    {
        return "servoy.SizeValidator";
    }

    public int[] getSupportedColumnTypes()
    {
        return new int[] { IColumnTypes.TEXT, IColumnTypes.MEDIA };
    }

    public void validate(Map props, Object arg) throws IllegalArgumentException
    {
        String propLength = (String)props.get(LENGTH_PROPERTY);
        int length = new Double(propLength.replace(',', '.')).intValue();
        if (arg instanceof byte[] && ((byte[])arg).length > length)
        {
            throw new IllegalArgumentException();
        }
        else if (arg instanceof String && ((String)arg).length() > length)
        {
            throw new IllegalArgumentException();
        }
    }
}
```