MyFaces Commons Converters

MyFaces Commons Converters contains some useful converters not provided by the spec, and a base class for create custom converters using myfaces-builder-plugin.

This base class (`org.apache.myfaces.commons.converter.ConverterBase`) allow use ValueExpressions and in 2.0 version provides an implementation of StateHelper. This is useful in cases where parameters for the converter are bound to ValueExpression instance that are evaluated at runtime. By default, converter parameters are evaluated on build time (the first time the view is built), so they cannot change over the time (further web postback requests).

To understand this, take a look at this example:

```
<h:form id="form1">
    <h:messages showDetail="false" showSummary="true" ></h:messages>
    <h:panelGrid columns="3">

        <h:outputLabel for="type" value="Insert a type" />
        <h:selectOneMenu id="type" immediate="true" value="#{numberBean.type}" >
            <f:selectItem itemLabel="number" itemValue="number"/>
            <f:selectItem itemLabel="currency" itemValue="currency"/>
            <f:selectItem itemLabel="percent" itemValue="percent"/>
        </h:selectOneMenu>
        <h:message for="type" styleClass="error" />

        <h:outputLabel for="number1" value="Insert a number" />
        <h:inputText id="number1" value="#{numberBean.numberMap['number1']}" required="true">
            <mcc:convertNumber destType="java.lang.Double" type="#{mc:findComponent('form1:type').value}"/>
        </h:inputText>
        <h:message for="number1" styleClass="error" />

        <h:commandButton id="validateButton"
         value="#{example_messages['button_submit']}"
         action="#{numberBean.submit}" />

    </h:panelGrid>
</h:form>
```

It is a `<h:selectOneMenu>` that allows you to choose a type that is then read by a converter. This can't work with `<f:convertNumber>`, because the ValueExpression on type is evaluated at build time. But in this case, `<mcc:convertNumber>` evaluate them each time the conversion occur. Note the `immediate="true"` in `<h:selectOneMenu>` makes the value of the component be evaluated before the conversion over the input text.

Also, in this project you can found:

- <mcc:convertBoolean> : Simple true/false converter with alternate representations of true false (yes/no), (1/0), and (way/no way).
- <mcc:convertDateTime> : Simple convert that overrides the spec DateTimeConverter and uses TimeZone.getDefault() as the base timezone, rather than GMT. Convert date time using normal system timezone like it should. In JSF 2.0 a new param for enable this behavior for f: convertDateTime was created called javax.faces.DATETIMECONVERTER_DEFAULT_TIMEZONE_IS_SYSTEM_TIMEZONE.
- <mcc:convertEnum> : Alternate enum converter.
- <mcc:convertNumber> : Converter which uses either the manually set destType or the value binding to determine the correct destination type to convert the number to This tag creates a number formatting converter and associates it with the nearest parent UIComponent. It uses either the manually set destType or the value binding to determine the correct destination type to convert the number to.