There are multiple converter in codify 3, and 2. PhoneConverter not found being used in xhtml of any project but still picked as done by me

1. Need there was special pattern that need to remove before saving to db. Prepare, RnD scritpt to explain
2. SSn converter need to explore it is used in chester. It can be used to explain in interview

Both converters are in

D:\WrkSpaces\wrks\_jsf\_codifyTest\JavaServerFaces

Need to see in use first then check code

<p:inputMask id="input" value="#{ctxBean.entitySearchForm.ssn}" styleClass=" cd-dt-secret" slotChar="\_" label="#{uilabel.customer\_search\_customer\_ssn}" >

<f:converter converterId="MaskConverter" />

<f:attribute name="mask" value="?999-99-9999"/>

<p:ajax event="change" process="@this" update=":pageHeaderForm:sessionDirtyFlagId msg "/>

</p:inputMask>

D:\src\Codify3\foundation\codify-web\src\main\java\com\thirdpillar\codify\foundation\jsf\converter\ PhoneConverter.java

/\*\* $URL$

\* $Id$

\*/

package com.thirdpillar.codify.foundation.jsf.converter;

import org.apache.commons.lang.StringUtils;

import java.util.regex.Matcher; import java.util.regex.Pattern;

import javax.faces.component.UIComponent;

import javax.faces.context.FacesContext;

import javax.faces.convert.Converter;

/\*\*

\* DOCUMENT ME!

\*

\* @author lokesh.rajput

\* @version 1.0

\* @since 1.0

\*/

public class PhoneConverter implements Converter {

//~ Static fields/initializers -------------------------------------------------------------------------------------

private static final String PHONE\_REGEX = "([\\(])?\\d{3}([\\)])?\\d{3}([-])?\\d{4}";

private static final String PHONE\_REGEX\_WRONG1 = "[\\(]\\d{3}\\d{3}([-])?\\d{4}";

private static final String PHONE\_REGEX\_WRONG2 = "\\d{3}[\\)]\\d{3}([-])?\\d{4}";

//~ Methods --------------------------------------------------------------------------------------------------------

public Object getAsObject(FacesContext context, UIComponent component, String value) {

value = StringUtils.trim(value);

if (StringUtils.isEmpty(value)) {

return null;

}

if (!matched(value, PHONE\_REGEX)) {

return null;

} else {

if (matched(value, PHONE\_REGEX\_WRONG1) || matched(value, PHONE\_REGEX\_WRONG2)) {

return null;

}

}

String[] phoneComps = StringUtils.split(value, "()-");

String phone = StringUtils.join(phoneComps);

return phone;

}

public String getAsString(FacesContext context, UIComponent component, Object value) {

String phone = (String) value;

if(StringUtils.isEmpty(phone) || phone.length() != 10){

return "";

}

phone = "(" + phone.substring(0, 3) + ")" + phone.substring(3, 6) + "-" + phone.substring(6);

return phone;

}

private boolean matched(String value, String pattern) {

boolean matched = false;

Pattern mask = Pattern.compile(pattern);

Matcher matcher = mask.matcher(value);

matched = matcher.matches();

return matched;

}

}

Custom converter in JSF 2.0

By [mkyong](https://www.mkyong.com/author/mkyong/" \o "mkyong) | November 1, 2010 | Viewed : 170,470 times +334 pv/w

In this article, we show you how to create a custom converter in JSF 2.0.

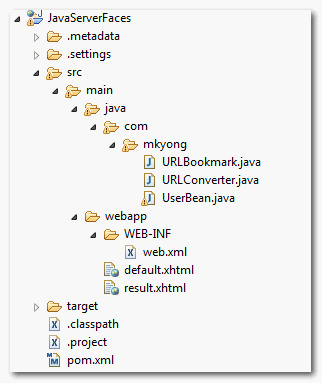
*Steps*  
1. Create a converter class by implementing **javax.faces.convert.Converter** interface.  
2. Override both **getAsObject()** and **getAsString()** methods.  
3. Assign an unique converter ID with **@FacesConverter** annotation.  
4. Link your custom converter class to JSF component via **f:converter** tag.

Custom converter example

A detail guide to create a JSF 2 custom converter name “URLConverter”, which is used to convert a String into an URL format (add HTTP protocol in front only :)) and store it into an Object.

1. Folder Structure

Folder structure of this example.



2. Converter class

Create a custom converter class by implementing **javax.faces.convert.Converter** interface.

package com.mkyong;

import javax.faces.convert.Converter;

public class URLConverter implements Converter{

//...

}

Override following two methods :  
**1. getAsObject()**, converts the given string value into an Object.  
**2. getAsString()**, converts the given object into a String.

public class URLConverter implements Converter{

@Override

public Object getAsObject(FacesContext context, UIComponent component,

String value) {

//...

}

@Override

public String getAsString(FacesContext context, UIComponent component,

Object value) {

//...

}

Assign an converter ID with **@FacesConverter** annotation.

package com.mkyong;

import javax.faces.convert.Converter;

@FacesConverter("com.mkyong.URLConverter")

public class URLConverter implements Converter{

//...

}

See full custom converter source code :

**URLConverter.java**

package com.mkyong;

import javax.faces.application.FacesMessage;

import javax.faces.component.UIComponent;

import javax.faces.context.FacesContext;

import javax.faces.convert.Converter;

import javax.faces.convert.ConverterException;

import javax.faces.convert.FacesConverter;

import org.apache.commons.validator.UrlValidator;

@FacesConverter("com.mkyong.URLConverter")

public class URLConverter implements Converter{

@Override

public Object getAsObject(FacesContext context, UIComponent component,

String value) {

String HTTP = "http://";

StringBuilder url = new StringBuilder();

//if not start with http://, then add it

if(!value.startsWith(HTTP, 0)){

url.append(HTTP);

}

url.append(value);

//use Apache common URL validator to validate URL

UrlValidator urlValidator = new UrlValidator();

//if URL is invalid

if(!urlValidator.isValid(url.toString())){

FacesMessage msg =

new FacesMessage("URL Conversion error.",

"Invalid URL format.");

msg.setSeverity(FacesMessage.SEVERITY\_ERROR);

throw new ConverterException(msg);

}

URLBookmark urlBookmark = new URLBookmark(url.toString());

return urlBookmark;

}

@Override

public String getAsString(FacesContext context, UIComponent component,

Object value) {

return value.toString();

}

}

In this custom converter class, it’s given a converter id as “**com.mkyong.URLConverter**“, and converts any given String (by adding a “http” in front) into “**URLBookmark**” object.

In addition, if URL validation is failed, return a **FacesMessage** object with declared error message.

**URLBookmark.java**

package com.mkyong;

public class URLBookmark{

String fullURL;

public URLBookmark(String fullURL) {

this.fullURL = fullURL;

}

public String getFullURL() {

return fullURL;

}

public void setFullURL(String fullURL) {

this.fullURL = fullURL;

}

public String toString(){

return fullURL;

}

}

3. Managed Bean

A normal managed bean named “user”, nothing special here.

package com.mkyong;

import java.io.Serializable;

import javax.faces.bean.ManagedBean;

import javax.faces.bean.SessionScoped;

@ManagedBean(name="user")

@SessionScoped

public class UserBean implements Serializable{

String bookmarkURL;

public String getBookmarkURL() {

return bookmarkURL;

}

public void setBookmarkURL(String bookmarkURL) {

this.bookmarkURL = bookmarkURL;

}

}

4. JSF Page

Link above custom converter to JSF component via “**converterId**” attribute in “**f:converter**” tag.

**default.xhtml**

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://java.sun.com/jsf/html"

xmlns:f="http://java.sun.com/jsf/core"

>

<h:body>

<h1>Custom converter in JSF 2.0</h1>

<h:form>

<h:panelGrid columns="3">

Enter your bookmark URL :

<h:inputText id="bookmarkURL" value="#{user.bookmarkURL}"

size="20" required="true" label="Bookmark URL">

<f:converter converterId="com.mkyong.URLConverter" />

</h:inputText>

<h:message for="bookmarkURL" style="color:red" />

</h:panelGrid>

<h:commandButton value="Submit" action="result" />

</h:form>

</h:body>

</html>

**result.xhtml**

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://java.sun.com/jsf/html"

xmlns:f="http://java.sun.com/jsf/core"

xmlns:c="http://java.sun.com/jsp/jstl/core"

>

<h:body>

<h1>Custom converter in JSF 2.0</h1>

<h:panelGrid columns="2">

Bookmark URL :

<h:outputText value="#{user.bookmarkURL}" />

</h:panelGrid>

</h:body>

</html>

5. Demo

Enter a valid URL, without “http”.



Add a “http” in front of the valid URL and display it.



If an invalid URL is provided, return the declared error message.



Download Source Code

Download It – [JSF-2-Custom-Converter-Example.zip](http://www.mkyong.com/wp-content/uploads/2010/11/JSF-2-Custom-Converter-Example.zip) (11KB)