

The BalusC Code

Code depot of a Java EE developer

Ad by Google

Upload

JSF Tutorial

Upload Files Web

Java JSF

Sunday, December 27, 2009

Uploading files with JSF 2.0 and Servlet 3.0

Fast menu

Introduction

The new [Servlet 3.0 specification](#) made uploading files [really easy](#). However, because JSF 2.0 isn't initially designed to be primarily used on top of Servlet 3.0 and should be backwards compatible with Servlet 2.5, it lacks a standard file upload component. Until now you could have used among others Tomahawk's [t:inputFileUpload](#) for that. But as of now (December 2009) Tomahawk appears not to be "JSF 2.0 ready" yet and has problems here and there when being used on a JSF 2.0 environment. When you're targeting a Servlet 3.0 compatible container such as Glassfish v3, then you could also just create a custom JSF file upload component yourself.

To prepare, you need to have a `Filter` which puts the parts of a `multipart/form-data` request into the request parameter map before the `FacesServlet` kicks in. The `FacesServlet` namely doesn't have builtin facilities for this relies on the availability of the submitted input component values in the request parameter map. [You can find it all here](#). Put the three classes `MultipartFileMap`, `MultipartFileFilter` and `MultipartFileRequest` in the classpath. The renderer of the custom file upload component relies on them in case of `multipart/form-data` requests.

[Back to top](#)

Custom component and renderer

With the new JSF 2.0 annotations it's now more easy to create custom components yourself. You don't need to hassle with somewhat opaque XML configurations anymore. I however only had a little hard time in figuring the best way to create custom components with help of annotations, because it's nowhere explained in the [Java EE 6 tutorial](#) nor the [JSR314 - JSF 2.0 Specification](#). I am sure that the Sun JSF guys are also reading here, so here it is: *Please work on that, it was already opaque in JSF 1.x and it should not be that more opaque in JSF 2.0!*

At any way, I finally figured it with little help of [Jim Driscoll's blog](#) and exploring the JSF 2.0 source code.

First, let's look what we need: in the line of `h:inputText` component which renders a `HTML input type="text"` element, we would like to have a fictive `h:inputFile` component which renders a `HTML input type="file"` element. As `h:inputText` component is represented by a [HtmlInputText](#) class, we would thus like to have a `HtmlInputFile` class which extends `HtmlInputText` and overrides the `rendererType` so that it generates a `HTML input type="file"` element instead.

Okay, that's not big deal, so here it is:

```
/*
 * net/balusc/jsf/component/html/HtmlInputFile.java
 *
 * Copyright (C) 2009 BalusC
 *
 * This program is free software: you can redistribute it and/or modify it under the terms of the
 * GNU Lesser General Public License as published by the Free Software Foundation, either version 3
 * of the License, or (at your option) any later version.
 *
 * This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without
 * even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public License along with this library.
 * If not, see <http://www.gnu.org/licenses/>.
 */

package net.balusc.jsf.component.html;

import javax.faces.component.FacesComponent;
import javax.faces.component.html.HtmlInputText;

/**
 * Faces component for <code>input type="file"</code> field.
 *
 * @author BalusC
 * @link http://balusc.blogspot.com/2009/12/uploading-files-with-jsf-20-and-servlet.html
 */
@FacesComponent(value = "HtmlInputFile")
public class HtmlInputFile extends HtmlInputText {

    // Getters -----

    @Override
    public String getRendererType() {
        return "javax.faces.File";
    }

}
```

The nice thing is that this component inherits all of the standard attributes of `HtmlInputText` so that you don't need to redefine them (*fortunately* not; it would have been a fairly tedious task and a lot of code).

The value of the [@FacesComponent](#) annotation represents the **component-type** which is to be defined in the taglib XML file (shown later). The `getRendererType()` should return the **renderer-type** of the renderer class which is to be annotated using [@FacesRenderer](#).

Extending the `renderer` is however quite a work when you want to be implementation independent, you need to take all possible attributes into account here as well. In this case we assume that you're going to use and stick to [Mojarra 2.x](#) forever (and thus not replace by another JSF implementation such as `MyFaces` sooner or later). Analogous with extending `HtmlInputText` to `HtmlInputFile` we thus want to extend its `Mojarra-specific` `TextRenderer` to `FileRenderer` so that it renders a `HTML input type="file"` element instead.

ABOUT



[BAUKE SCHOLTZ](#)

[View my complete profile](#)

DONATE

For the ones who want to express their excessive thanks for my work, I used to have an Amazon wishlist linked on my [stackoverflow.com](#) profile, but they unfortunately don't send anything else than books to Curaçao and right now I don't have any interesting books on the list anymore (to anyone who've sent books before: thank you very much, I got 6 books in 6 months). You can always donate something so that I can use it for other stuff which Amazon doesn't send, such as Nespresso coffee.

[Donate](#)

TAGS

ActionListener Ajax Authentication CDI
Communication Composite Component
Converter CSV Custom Component
DAO DataTable Design Patterns
Download File DTO Dutch
Eclipse EJB Exception-Handling
Faces Filter Focus Glassfish
Highlight HTML5 i18n Immediate Include
JavaScript JDBC JPA JSF
JSF2 JSP Managed
Bean Messages MySQL
OmniFaces Performance
PhaseListener POST-Redirect-GET
PrimeFaces Rant Renderer
SelectBooleanCheckbox SelectOneMenu
Servlet Shiro StackOverflow
TabbedPanel Tomahawk Tomcat
Tutorial Unicode Upload File
UseBean Utility Validator
ValueChangeListener Vldoc
ViewScoped Whitespace

ARTICLES

- 3000 (1)
- 2013 (3)
- 2012 (12)
- 2011 (2)
- 2010 (10)
- ▼ 2009 (7)
 - ▼ December (2)
 - [Uploading files with JSF 2.0 and Servlet 3.0](#)
 - [Uploading files in Servlet 3.0](#)
 - September (2)
 - May (2)
 - February (1)
- 2008 (7)
- 2007 (19)
- 2006 (22)

FAVORITES

[Use the right keywords](#)
[How to write good code](#)
[How to write bad code](#)
[Java SE 7.0 API documentation](#)
[Java EE 6.0 API documentation](#)
[Servlet 3.0 specification \(JSR315\)](#)
[JSP/EL 2.2 specification \(JSR245\)](#)
[JSP/Servlet tutorials on Coreservlets](#)
[JSTL 1.2 specification \(JSR052\)](#)
[JSTL 1.1 TLD documentation](#)
[JSF RI \(Mojarra\) home page](#)
[JSF 2.0 specification \(JSR314\)](#)

```

/*
 * net/balusc/jsf/renderer/html/FileRenderer.java
 *
 * Copyright (C) 2009 BalusC
 *
 * This program is free software: you can redistribute it and/or modify it under the terms of the
 * GNU Lesser General Public License as published by the Free Software Foundation, either version 3
 * of the License, or (at your option) any later version.
 *
 * This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without
 * even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public License along with this library.
 * If not, see <http://www.gnu.org/licenses/>.
 */

package net.balusc.jsf.renderer.html;

import java.io.File;
import java.io.IOException;

import javax.faces.component.UIComponent;
import javax.faces.component.UIInput;
import javax.faces.context.FacesContext;
import javax.faces.context.ResponseWriter;
import javax.faces.convert.ConverterException;
import javax.faces.render.FacesRenderer;

import net.balusc.http.multipart.MultipartRequest;

import com.sun.faces.renderkit.Attribute;
import com.sun.faces.renderkit.AttributeManager;
import com.sun.faces.renderkit.RenderKitUtils;
import com.sun.faces.renderkit.html_basic.TextRenderer;

/**
 * Faces renderer for <code>input type="file"</code> field.
 *
 * @author BalusC
 * @link http://balusc.blogspot.com/2009/12/uploading-files-with-jsf-20-and-servlet.html
 */
@FacesRenderer(componentFamily = "javax.faces.Input", rendererType = "javax.faces.File")
public class FileRenderer extends TextRenderer {

    // Constants -----
    private static final String EMPTY_STRING = "";
    private static final Attribute[] INPUT_ATTRIBUTES =
        AttributeManager.getAttributes(AttributeManager.Key.INPUTTEXT);

    // Actions -----

    @Override
    protected void getEndTextToRender(
        (FacesContext context, UIComponent component, String currentValue)
        throws IOException
    ) {
        ResponseWriter writer = context.getResponseWriter();
        writer.startElement("input", component);
        writer.writeAttributeIfNecessary(context, writer, component);
        writer.writeAttribute("type", "file", null);
        writer.writeAttribute("name", (component.getClientId(context)), "clientId");

        // Render styleClass, if any.
        String styleClass = (String) component.getAttributes().get("styleClass");
        if (styleClass != null) {
            writer.writeAttribute("class", styleClass, "styleClass");
        }

        // Render standard HTML attributes expect of styleClass.
        RenderKitUtils.renderPassThruAttributes(
            context, writer, component, INPUT_ATTRIBUTES, getNonOnChangeBehaviors(component));
        RenderKitUtils.renderXHTMLStyleBooleanAttributes(writer, component);
        RenderKitUtils.renderOnChange(context, component, false);

        writer.endElement("input");
    }

    @Override
    public void decode(FacesContext context, UIComponent component) {
        rendererParamsNotNull(context, component);
        if (!shouldDecode(component)) {
            return;
        }
        String clientId = decodeBehaviors(context, component);
        if (clientId == null) {
            clientId = component.getClientId(context);
        }
        File file = ((MultipartRequest) context.getExternalContext().getRequest()).getFile(clientId);

        // If no file is specified, set empty String to trigger validators.
        ((UIInput) component).setSubmittedValue((file != null) ? file : EMPTY_STRING);
    }

    @Override
    public Object getConvertedValue(FacesContext context, UIComponent component, Object submittedValue)
        throws ConverterException
    {
        return (submittedValue != EMPTY_STRING) ? submittedValue : null;
    }
}

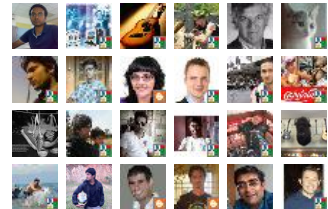
```

[JSF 2.0 PDL documentation](#)
[JSF 2.1 VDL documentation](#)
[JSF tutorial \(Java EE tutorial part II chpt 4 and on\)](#)
[JSF tutorials on Coreservlets](#)
[JSF 2.0: The Complete Reference \(book\)](#)
[Eclipse IDE for Java EE developers](#)
[Eclipse Video Tutorials](#)
[JBoss Tools for Eclipse](#)
[HTML Doctype explained](#)

FOLLOWERS

Join this site
 with Google Friend Connect

Members (634) [More »](#)



Already a member? [Sign in](#)

Note that the `@FacesRenderer` annotation also specifies a component family of `"javax.faces.Input"` and that this is nowhere specified in our `HtmlInputFile`. That's also not needed, it's already inherited from `HtmlInputText`.

Now, to use the custom JSF 2.0 file upload component in Facelets we really need to define another XML file. It's however not a big deal. You fortunately don't need to define all the tag attributes as you should have done in case of JSP. Just define the **namespace** (which you need to specify in the `xmlns` attribute of the `<html>` tag), the **tag name** (to identify the tag in XHTML) and the **component type** (as defined in the `@FacesComponent` of the associated component class).

Create a new XML file at `/WEB-INF/balusc.taglib.xml` and fill it as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<facelet-taglib
  xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-facelettaglibrary_2_0.xsd"
  version="2.0">

  <namespace>http://balusc.net/jsf/html</namespace>
  <tag>
    <tag-name>inputFile</tag-name>
    <component>
      <component-type>HtmlInputFile</component-type>
    </component>
  </tag>
</facelet-taglib>
```

You need to familiarize Facelets with the new taglib in `web.xml` as follows:

```
<context-param>
  <param-name>javax.faces.FACELETS_LIBRARIES</param-name>
  <param-value>/WEB-INF/balusc.taglib.xml</param-value>
</context-param>
```

Note, if you have multiple Facelets taglibs, then you can separate the paths with a semicolon ; .

[Back to top](#)

Basic use example

Here is a basic use example of a JSF managed bean and a Facelets page which demonstrates the working of all of the stuff.

First the managed bean **UploadBean**:

```
package net.balusc.example.upload;

import java.io.File;
import java.util.Arrays;

import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;

@ManagedBean
@RequestScoped
public class UploadBean {

    private String text;
    private File file;
    private String[] check;

    public void submit() {
        // Now do your thing with the obtained input.
        System.out.println("Text: " + text);
        System.out.println("File: " + file);
        System.out.println("Check: " + Arrays.toString(check));
    }

    public String getText() {
        return text;
    }

    public File getFile() {
        return file;
    }

    public String[] getCheck() {
        return check;
    }

    public void setText(String text) {
        this.text = text;
    }

    public void setFile(File file) {
        this.file = file;
    }

    public void setCheck(String[] check) {
        this.check = check;
    }
}
```

And now the Facelets page `upload.xhtml`:

```
<!DOCTYPE html>
<html
  xmlns="http://www.w3.org/1999/xhtml"
  xmlns:f="http://java.sun.com/jsf/core"
  xmlns:h="http://java.sun.com/jsf/html"
  xmlns:hh="http://balusc.net/jsf/html">
  <h:head>
    <title>JSF 2.0 and Servlet 3.0 file upload test</title>
    <style>label { float: left; display: block; width: 75px; }</style>
  </h:head>
  <h:body>
    <h:form id="form" method="post" enctype="multipart/form-data">
      <h:outputLabel for="text">Text: </h:outputLabel>
      <h:inputText id="text" value="#{uploadBean.text}" />
      <br />
      <h:outputLabel for="file">File: </h:outputLabel>
      <h:inputFile id="file" value="#{uploadBean.file}" />
      <h:outputText value="File #{uploadBean.file.name} successfully uploaded!"
        rendered="#{(not empty uploadBean.file)}" />
      <br />
      <h:selectManyCheckbox id="check" layout="pageDirection" value="#{uploadBean.check}">
        <f:selectItem itemLabel="Check 1:" itemValue="check1" />
        <f:selectItem itemLabel="Check 2:" itemValue="check2" />
      </h:selectManyCheckbox>
```

```
<h:commandButton value="submit" action="#{uploadBean.submit}" />
<h:messages />
</h:form>
</h:body>
</html>
```

Copy'n'paste the stuff and run it at <http://localhost:8080/playground/upload.jsf> (assuming that your local development server runs at port 8080 and that the context root of your playground web application project is called 'playground' and that you have the FacesServlet in web.xml mapped on *.jsf) and see it working! And no, you don't need to do anything with faces-config.xml, the managed bean is automatically found and initialized with help of the new JSF 2.0 annotations.

Note: this all is developed and tested with Eclipse 3.5 and Glassfish v3.

[Back to top](#)

Validate uploaded file

The lack of the support of @MultiPartConfig annotation in the filter and JSF also implies that the size of the uploaded file can't be restricted by the maxFileSize annotation field. It is however possible to attach a simple validator to the custom component. Here's an example:

```
package net.balusc.example.upload;

import java.io.File;

import javax.faces.application.FacesMessage;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.validator.FacesValidator;
import javax.faces.validator.Validator;
import javax.faces.validator.ValidatorException;

@FacesValidator(value = "fileValidator")
public class FileValidator implements Validator {

    private static final long MAX_FILE_SIZE = 10485760L; // 10MB.

    @Override
    public void validate(FacesContext context, UIComponent component, Object value)
        throws ValidatorException {
        {
            File file = (File) value;
            if (file != null && file.length() > MAX_FILE_SIZE) {
                file.delete(); // Free resources!
                throw new ValidatorException(new FacesMessage(String.format(
                    "File exceeds maximum permitted size of %d bytes.", MAX_FILE_SIZE)));
            }
        }
    }
}
```

You can attach it as follows:

```
<h:inputFile validator="fileValidator" />
```

You can also use a f:validator instead:


```
<h:inputFile>
  <f:validator validatorId="fileValidator" />
</h:inputFile>
```

That should be it. Also no faces-config stuff is needed here thanks to the annotations.

[Back to top](#)

Copyright - [GNU Lesser General Public License](#)

(C) December 2009, BalusC

Geplaatst door Bauke Scholtz op [5:35 PM](#) 

+1 [Recommend this on Google](#)

Labels: [Custom Component](#), [Facelets](#), [JSF](#), [JSF2](#), [Renderer](#), [Upload File](#)

97 comments:



Anthony said...

You beat me to it! Thanks a lot for all your hard work.

December 28, 2009 at 1:29 AM



Anthony said...

Would it make more sense to implement the max size check in the MultipartMap and have the size be configurable in the Filter?

December 28, 2009 at 11:34 AM



Dominik Dorn said...

balusC, you rock!
I've been looking for this for weeks, even filled complaints about the lacking fileUpload support in the api just to now find this post! Well done!

December 28, 2009 at 11:59 AM



BalusC said...

@Anthony: not if you would like to take benefit of JSF validation (and localized messaging!). Doing so in the filter would cause an exception which goes beyond the scope of JSF. I.e. user may face a HTTP 500 error instead.

@Dominik: no problem :)

December 28, 2009 at 3:38 PM

Dominik Dorn said...

@BalusC: hmm.. the filter breaks the required="true" attribute & the filters of the inputText components... do you have an idea whats going wrong?.

December 28, 2009 at 4:09 PM

Dominik Dorn said...

i mean validators.

December 28, 2009 at 4:15 PM

Anthony said...

Good point. I'll stick it in the request and pull it in through the Validator through the context. That should work for now.

December 28, 2009 at 4:46 PM

Dominik Dorn said...

Hey BalusC, I've created a full taglib out of your code, posted it on github and showed how to integrate it into a maven project on my blog <http://dominikdorn.com/?p=169>

December 28, 2009 at 7:13 PM

BalusC said...

@Dominik: you were right about the validators. I've updated the MultipartMap and FileRenderer for that. BufferedReader#readLine() returns null in case of empty strings while that should not be the case. Also, JSF ignores null values in the "required" validator, but it will convert empty strings back to null after the validation phase.

December 28, 2009 at 8:58 PM

Anthony said...

What are your guys's thoughts on creating a java.net project for this? It would be nice drop in solution for others.

December 29, 2009 at 7:37 AM

Carlos Raygoza said...

Hello BalusC,
I have this problem when i'm trying to run the program

StandardWrapperValve[Faces Servlet]: PWC1406: Servlet.service() for servlet Faces Servlet threw exception
java.io.IOException: El nombre de archivo, directorio o etiqueta del volumen no es válido

"the directory is not valid" and i've already created the folder upload in my project.
could you help me?
best regards
Carlos

January 17, 2010 at 12:12 PM

BalusC said...

The upload folder is (and should) not be relative to the web project, but to the local disk file system. Thus, if you're on Windows and the server is installed on C: disk, then "/upload" is basically "C:/upload". You should not store uploaded files in web project folders. They will get lost whenever you redeploy.

January 17, 2010 at 4:31 PM

Reiner said...

I used your classes to implement a file upload in my current web application project. And it worked fine. However, my dialogs stopped working.

I have a vast set of dialogs based on a facelet template and corresponding dialog client templates. These dialogs are dynamically included in a page template in a page template client.

So my assumption is that since I added your classes they changed the rendering chain in a way that the templates are not rendered correctly.

The process for a page should normally be:

- Load page client template
- Use page template for the page
- Load dialog client template
- Use the dialog template
- Render dialog by combining client template and template
- Render page by combining client template, includes and template

However, right now the flow looks like

- Load page client template
- Use page template for the page
- Load dialog client template
- Use the dialog template

- Render dialog << uses only the template but not the client template
- Render page by combining client template, includes and template

If you have any suggestion how this could happen or how I can find out why this is happening, do not hesitate to tell me ;-)

And by the way thanks for the fine work you've done on the file upload subject.

January 24, 2010 at 10:43 AM

Dominik Dorn said...

its a bug... currently I go around it with putting only the fileupload component in its own h:form and the rest of the page in another h:form... its dirty, but works (for now.)

however, its not the problem with the lifecycle, its just that the filter does not parse the other parameters send by the request correctly.

January 24, 2010 at 10:48 AM

Reiner said...

Forget about my post. The problem is, that I include two dialogs. While they are not both visible, the visibility control is done with f:subview tags. It looks like second time I include the same dialog it copies the setup for the first (which is incomplete).

January 24, 2010 at 12:38 PM

Squalphin said...

Thank you for your hard work! Your upload code really helped me out :)

January 25, 2010 at 5:02 AM

Carlos Raygoza said...

Hello BalusC,
First at all i want to thank you for your countribution and help us too.

i have a little doubt if i'm going to use a linux/unix server where can i change the path? it seems that this code is hardcoded to windows c: path..

Thanks for your time

best regards,

Carlos

February 13, 2010 at 2:00 PM

Cristian said...

Hello BalusC,

Congratulations on your work and thank you.

I'm using file upload functionality for JSF, but I am getting this error on Glassfish v3 :

```
java.lang.ClassCastException: org.apache.catalina.connector.RequestFacade cannot be cast to
jsf.file.upload.MultipartRequest
at jsf.file.upload.FileRenderer.decode(FileRenderer.java:63)
at javax.faces.component.UIComponentBase.decode(UIComponentBase.java:790)
at javax.faces.component.UIInput.decode(UIInput.java:744)
at javax.faces.component.UIComponentBase.processDecodes(UIComponentBase.java:1047)
at javax.faces.component.UIInput.processDecodes(UIInput.java:658)
at javax.faces.component.UIForm.processDecodes(UIForm.java:216)
at javax.faces.component.UIComponentBase.processDecodes(UIComponentBase.java:1042)
at javax.faces.component.UIViewRoot.processDecodes(UIViewRoot.java:941)
at com.sun.faces.lifecycle.ApplyRequestValuesPhase.execute(ApplyRequestValuesPhase.java:78)
at com.sun.faces.lifecycle.Phase.doPhase(Phase.java:101)
at com.sun.faces.lifecycle.LifecycleImpl.execute(LifecycleImpl.java:118)
at javax.faces.webapp.FacesServlet.service(FacesServlet.java:312)
at org.apache.catalina.core.StandardWrapper.service(StandardWrapper.java:1523)
at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:279)
at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:188)
at org.apache.catalina.core.StandardPipeline.invoke(StandardPipeline.java:641)
at com.sun.enterprise.web.WebPipeline.invoke(WebPipeline.java:97)
at com.sun.enterprise.web.PESessionLockingStandardPipeline.invoke(PESessionLockingStandardPipeline.java:85)
at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:185)
at org.apache.catalina.connector.CoyoteAdapter.doService(CoyoteAdapter.java:332)
at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:233)
at com.sun.enterprise.v3.services.impl.ContainerMapper.service(ContainerMapper.java:165)
at com.sun.grizzly.http.ProcessorTask.invokeAdapter(ProcessorTask.java:791)
at com.sun.grizzly.http.ProcessorTask.doProcess(ProcessorTask.java:693)
at com.sun.grizzly.http.ProcessorTask.process(ProcessorTask.java:954)
at com.sun.grizzly.http.DefaultProtocolFilter.execute(DefaultProtocolFilter.java:170)
at com.sun.grizzly.DefaultProtocolChain.executeProtocolFilter(DefaultProtocolChain.java:135)
at com.sun.grizzly.DefaultProtocolChain.execute(DefaultProtocolChain.java:102)
at com.sun.grizzly.DefaultProtocolChain.execute(DefaultProtocolChain.java:88)
at com.sun.grizzly.http.HttpProtocolChain.execute(HttpProtocolChain.java:76)
at com.sun.grizzly.ProtocolChainContextTask.doCall(ProtocolChainContextTask.java:53)
at com.sun.grizzly.SelectionKeyContextTask.call(SelectionKeyContextTask.java:57)
at com.sun.grizzly.ContextTask.run(ContextTask.java:69)
at com.sun.grizzly.util.AbstractThreadPool$Worker.doWork(AbstractThreadPool.java:330)
```

at com.sun.grizzly.util.AbstractThreadPool\$Worker.run(AbstractThreadPool.java:309)
at java.lang.Thread.run(Thread.java:619)
|#]

How can I solve this?

February 20, 2010 at 12:05 PM

Cristian said...

i just renamed the packages, i didn't modify your code

February 20, 2010 at 12:12 PM

BalusC said...

You need to have the MultipartFilter in the classpath.

February 20, 2010 at 1:52 PM

Cristian said...

hi BalusC,

I did add MultipartFilter to my classpath as such :

C:\...\app02\build\web\WEB-INF\classes\jsf\file\upload*.class

This doesn't work either :

C:\...\app02\build\web\WEB-INF\classes\jsf\file\upload\MultipartFilter.class

The same exception is thrown in the **decode** method of the **FileRenderer** class, at this line of code
File file = ((MultipartRequest) context.getExternalContext().getRequest()).getFile(clientId);

February 21, 2010 at 5:26 AM

TIM said...

hi BalusC

i have a new in jsf but my project need
upload file to server and specific path for upload
it can

February 23, 2010 at 12:08 AM

BalusC said...

Yes, you can. Just change the init-param for upload path in MultipartFilter accordingly.

February 23, 2010 at 9:57 AM

TIM said...

This comment has been removed by the author.

February 24, 2010 at 10:58 AM

TIM said...

This comment has been removed by the author.

February 24, 2010 at 11:04 AM

TIM said...

BalusC Thank you very much save a lot time
please suggest me for upload file and save to
blob field on Database

February 24, 2010 at 1:26 PM

TIM said...

can suggest me for download from
server via JSF 2

February 24, 2010 at 1:43 PM

Kiichiro Muto 武藤樹一郎 said...

This comment has been removed by the author.

February 25, 2010 at 3:29 AM

Peter Johansson said...

Thanks for this article, It was a little overkill but gave me some ideas of how to approach.

[File upload with JSF2 using base class](#)

/the ironic programmer

March 5, 2010 at 2:01 PM

Ivv said...

How can i open a seperate url using jsf

March 9, 2010 at 12:46 AM

kedves_uram said...

Hi BalusC,

Great job, thank you very much!

Could you please give me a hint on this message! I don't know what could be the error:

"Expression Error: Named Object: HtmlInputFile not found.

javax.faces.FacesException: Expression Error: Named Object: HtmlInputFile not found.

at com.sun.faces.application.ApplicationImpl.createComponentApplyAnnotations(ApplicationImpl.java:1809)

at com.sun.faces.application.ApplicationImpl.createComponent(ApplicationImpl.java:1090)

at javax.faces.application.Application.createComponent(Application.java:956)

at

com.sun.faces.facelets.tag.jsf.ComponentTagHandlerDelegateImpl.createComponent(ComponentTagHandlerDelegat
eImpl.java:326)

at

com.sun.faces.facelets.tag.jsf.ComponentTagHandlerDelegateImpl.apply(ComponentTagHandlerDelegateImpl.java:1
45)

at javax.faces.view.facelets.DelegatingMetaTagHandler.apply(DelegatingMetaTagHandler.java:114)

....and so on.."

I've created the taglib in the WEB-INF, and placed the

"

javax.faces.FACELETS_LIBRARIES

/WEB-INF/balusc.taglib.xml

"

in the web.xml

I would really appreciate if you'd help me!

thx,

kedves_uram

March 9, 2010 at 9:47 PM

Juana said...

Hello, how did you solve this problem:

java.lang.ClassCastException: org.apache.catalina.connector.RequestFacade cannot be cast to

jsf.file.upload.MultipartRequest.

April 7, 2010 at 6:30 PM

imie said...

Problem "java.lang.ClassCastException: org.apache.catalina.connector.RequestFacade cannot be cast to
jsf.file.upload.MultipartRequest"

Solution

In class FileRenderer find

"File file = ((MultipartRequest) context.getExternalContext().getRequest()).getFile(clientId);"

Replace by:

"File file = null;

if (context.getExternalContext().getRequest() instanceof MultipartRequest) {

file = ((MultipartRequest) context.getExternalContext().getRequest()).getFile(clientId);

}"

In *.xhtml

Don't forget about "h:form method="post" enctype="multipart/form-data"

May 26, 2010 at 10:51 AM

Brendan said...

Has anyone managed to get this working as a column in a datatable?

I get a ClassCastException when I click on a row in the table (which is selectable) that contains an
hh:inputFile component as a column. The component renders ok in the table but when you click on Browse...
nothing happens.

WARNING: org.apache.catalina.connector.RequestFacade cannot be cast to

uk.co.sportquest.jsfbeans.helper.MultipartRequest

java.lang.ClassCastException: org.apache.catalina.connector.RequestFacade cannot be cast to

uk.co.sportquest.jsfbeans.helper.MultipartRequest

at uk.co.sportquest.jsfbeans.helper.FileRenderer.decode(FileRenderer.java:87)

at javax.faces.component.UIComponentBase.decode(UIComponentBase.java:790)

at javax.faces.component.UIInput.decode(UIInput.java:744)

at javax.faces.component.UIComponentBase.processDecodes(UIComponentBase.java:1047)

at :619)

May 27, 2010 at 2:26 PM

BalusC said...

@Brendan: add `enctype="multipart/form-data"` to the h:form.

May 27, 2010 at 2:48 PM

Yusuf said...

Thank you BalusC, I am doing my first JSF project and it requires the file upload functionality.

Your post saved me A LOT of time. It is also the best solution I found. Now I can build on the base you provided.

Thanks to you I can meet my deadline!

I appreciate your time and effort.

May 29, 2010 at 4:59 AM

Joel said...

This comment has been removed by the author.

June 14, 2010 at 1:25 PM

Brendan said...

Hi, I'm just belatedly responding to your suggestion to use `<h:form enctype="multipart/form-data">`. I am doing this but had not made the edit suggested by imie.

Having done this the exception goes away but when in a cell in a PrimeFaces p:dataTable nothing happens. It works ok in a h:dataTable, so I guess I need to talk to the PrimeFaces people.

Great stuff anyway, Thanks.

June 18, 2010 at 7:20 AM

Brendan said...

Hi, I've got a form (`enctype=multipart/form-data`) with just the `hh:inputFile` component and a submit button. The submit button calls an action method in the backing bean which in this case generates an error. I add a `FacesMessage` and return an empty string to re-render the page. I then get this error, any ideas what this could be?

Many thanks,
Brendan.

```
SEVERE: Error Rendering View[/teamEdit.xhtml]
java.lang.IllegalArgumentException: This is a File field. Use #getFile() instead.
at uk.co.sportquest.jsfbeans.helper.MultipartMap.getParameter(MultipartMap.java:160)
at uk.co.sportquest.jsfbeans.helper.MultipartRequest.getParameter(MultipartRequest.java:69)
at com.sun.faces.context.RequestParameterMap.get(RequestParameterMap.java:71)
at com.sun.faces.context.RequestParameterMap.get(RequestParameterMap.java:52)
at com.sun.faces.context.BaseContextMap$EntryIterator.next(BaseContextMap.java:253)
at com.sun.faces.context.BaseContextMap$EntryIterator.next(BaseContextMap.java:236)
```

June 23, 2010 at 2:30 PM

Carlos E. R. Diógenes said...

Hi BalusC,

Thanks for this great work. With this I'm able to upload files withing JSF, but I can't use `@ConversationScoped` to hold the uploaded file. The error that I get is similar to the one described here: <http://seamframework.org/Community/ConversationScopeNotWorking#comment104493>. The conversation just go away after I submit the form with the `enctype="multipart/form-data"`. Do you have any idea why this happen? I'm still learning the JSF life cycle and until now I have no idea what could be wrong.

Best regards.

June 29, 2010 at 3:25 PM

Alex said...

First of all thank your for sharing your experience in this blog! Your article about using JSF 2.0 and Servlets 3.0 for file uploading is quite amazing. I really like the idea of encapsulating code into taglibs to get a modular structure into JSF projects. I am quite a fan of this technique and so I am using taglibs very often.

Currently I am looking for an acceptable solution to upload files in a JSF2.0 web application. I tried to run your code but the Multipart issue is twisting my brain. First of all the UploadBean action "submit()" is never called. The method is only invoked if I remove the `enctype="multipart/form-data"` from the h:form tag.

The second issue is about these lines of code in the MultipartMap class:

```
for (Part part : multipartRequest.getParts()) {
    String filename = getFilename(part);
```

```
if (filename == null) {  
    processTextPart(part);  
} else if (!filename.isEmpty()) {  
    processFilePart(part, filename);  
}  
}
```

For every multipart encoded form that is submitted the method `multipartRequest.getParts()` returns null.

Any ideas what could be wrong?

I am using Mojarra JSF 2.0.3 and Tomcat 7 with Servlet 3.0.

Kind regards
Alex

August 5, 2010 at 1:51 PM

BalusC said...

It turns out that the current Tomcat 7 beta strictly requires the `HttpServletRequest#getParts()` to be called inside a servlet with `@MultipartConfig`, not inside a filter!

I reported [issue 49711](#) to the Tomcat guys. Hope they get it fixed as well.

August 5, 2010 at 4:43 PM

Alex said...

Thanks BalusC.

I was guessing something like that. I wrote a servlet instead, now the returned Collection of `getParts` is no longer empty.

Thanks for the support... that saved my weekend!

August 6, 2010 at 1:20 PM

ThomasF said...

Hi BalusC,

thx for the implementation. Is it possible to get the `MultipartMap` handling hidden-fields as well? They are ignored yet. Maybe they are not returned from `getParts()` ...

Greetings, Thomas

August 9, 2010 at 2:20 PM

coubeatczech said...

Hi, I tried it, to run on my glassfishv3 and it doesn't run. When I click on the submit button, nothing really happens. I just cypasted it, I didn't do any changes.....:t

August 21, 2010 at 4:58 PM

vogdb said...

Hi, guys. Please hint me with a link or something. How I can pack all these files in one jar file and use it in other projects (most JSF2 projects) just as icefaces or primefaces. I've tried various ways, but stuck. My problem very close to this [on the jsf sun forum](#)

P.S. Thanks for cool upload. It really helpful article.

August 22, 2010 at 5:09 AM

Jens said...

Hey Alex,

I am facing the same problem that you had.

Could you be so nice to post your servlet ?

Thanks a lot !
Jens

September 2, 2010 at 10:09 AM

Alex said...

Hey Jens,

sorry for answering so late! I have been very busy since the end of the summer. :)

To get straight to the point: Because of performance related issues I stopped implementing my file upload using the Tomcat Servlet API 3.0. The huge disadvantage of `HttpServletRequest#getParts()` is, that you do not have any chance to grab the `InputStream` of a incoming file. The whole file is uploaded to a temporary directory. After that you can start working with the file. For my purposes I needed an `InputStream` to pass the received data through a business API into an application server environment.

To cut a long story short: My application takes the `ServletInputStream` within the `HttpServlet#doPost` method and reads out the `MultiPart` related data with some code that I wrote myself. If you are interested how something like that can be implemented, as I explained in short terms, without using Servlet API 3.0, please let me know.

Concerning the approach feeding Servlet API 3.0 with your file upload data, just use BalusC's MultiPartMap (<http://balusc.blogspot.com/2009/12/uploading-files-in-servlet-30.html#MoreAbstraction>) and take a servlet instad of the filter. You can use RequestDispatcher#forward(request, response) to pass the uploaded file to a HttpServlet like the one in BalusC's Filter example.

Regards Alex

October 8, 2010 at 9:06 AM

Family said...

This comment has been removed by the author.

February 16, 2011 at 5:49 AM

soumitra said...

In '

' error : "value" attribute is not found.

March 10, 2011 at 3:10 AM

soumitra said...

error: 'value' attribute is not found.

March 10, 2011 at 3:10 AM

Dave Squire said...

Thank you, BalusC, for reporting [Issue 49711](#) to the Tomcat developers.

They have fixed this issue as of Tomcat 7.0.7. But, you will have to add an attribute to the Context element of your application's context.xml file in order for the fix to work. The attribute is allowCasualMultipartParsing and it should be set to "true". For example:

April 12, 2011 at 4:03 PM

D said...

Hi balusC,

I want to upload images with JSF 2.0 with facelets not servlets. Can i do so using ur code provided here?

April 13, 2011 at 2:29 PM

Victor Manuel said...

Hi BalusC, i ´m having a problem with this part of the multipartmap

```
for (Part part : multipartRequest.getParts()) {  
    String filename = getFilename(part);  
    if (filename == null) {  
        processTextPart(part);  
    } else if (!filename.isEmpty()) {  
        processFilePart(part, filename);  
    }  
}
```

multipartRequest.getParts is null. I ´m using jboss 6 and jsf the element is inside a xhtml.

On the other hand, if a use quick help control+spc inside the hh:inputfile> element no value parameter is shown, is that normal¿? thanks!

April 25, 2011 at 9:09 AM

Ashwin said...

```
WARNING: StandardWrapperValve[Faces Servlet]: PWC1406: Servlet.service() for servlet Faces Servlet threw exception  
java.io.IOException: The system cannot find the path specified  
at java.io.WinNTFileSystem.createFileExclusively(Native Method)  
at java.io.File.checkAndCreate(File.java:1704)  
at java.io.File.createTempFile(File.java:1792)  
at net.balusc.http.multipart.MultipartMap.processFilePart(MultipartMap.java:187)  
at net.balusc.http.multipart.MultipartMap.(MultipartMap.java:71)  
at net.balusc.http.multipart.MultipartMap.(MultipartMap.java:51)  
at net.balusc.http.multipart.MultipartRequest.(MultipartRequest.java:40)  
at net.balusc.http.multipart.MultipartFilter.doFilter(MultipartFilter.java:49)  
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:256)  
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:215)  
at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:279)  
at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:175)  
at org.apache.catalina.core.StandardPipeline.doInvoke(StandardPipeline.java:655)  
at org.apache.catalina.core.StandardPipeline.invoke(StandardPipeline.java:595)  
at com.sun.enterprise.web.WebPipeline.invoke(WebPipeline.java:98)  
at com.sun.enterprise.web.PESessionLockingStandardPipeline.invoke(PESessionLockingStandardPipeline.java:91)  
at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:162)  
at org.apache.catalina.connector.CoyoteAdapter.doService(CoyoteAdapter.java:326)  
at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:227)  
at com.sun.enterprise.v3.services.impl.ContainerMapper.service(ContainerMapper.java:170)  
at com.sun.grizzly.http.ProcessorTask.invokeAdapter(ProcessorTask.java:822)
```

```
at com.sun.grizzly.http.ProcessorTask.doProcess(ProcessorTask.java:719)
at com.sun.grizzly.http.ProcessorTask.process(ProcessorTask.java:1013)
at com.sun.grizzly.http.DefaultProtocolFilter.execute(DefaultProtocolFilter.java:225)
at com.sun.grizzly.DefaultProtocolChain.executeProtocolFilter(DefaultProtocolChain.java:137)
at com.sun.grizzly.DefaultProtocolChain.execute(DefaultProtocolChain.java:104)
at com.sun.grizzly.DefaultProtocolChain.execute(DefaultProtocolChain.java:90)
at com.sun.grizzly.http.HttpProtocolChain.execute(HttpProtocolChain.java:79)
at com.sun.grizzly.ProtocolChainContextTask.doCall(ProtocolChainContextTask.java:54)
at com.sun.grizzly.SelectionKeyContextTask.call(SelectionKeyContextTask.java:59)
at com.sun.grizzly.ContextTask.run(ContextTask.java:71)
at com.sun.grizzly.util.AbstractThreadPool$Worker.doWork(AbstractThreadPool.java:532)
at com.sun.grizzly.util.AbstractThreadPool$Worker.run(AbstractThreadPool.java:513)
at java.lang.Thread.run(Thread.java:619)
```

May 10, 2011 at 3:44 AM

Ashwin said...

i got this error beacuse of temp file creation.i am using windows 7 and glassfish server with eclipse

May 10, 2011 at 3:46 AM

Mark said...

Got this code working, but Netbeans (6.9.1) claims that in the line:

The attribute value is not defined in the component's interface. Convert to a composite component

Any ideas?

May 11, 2011 at 12:21 PM

vegetatrux said...

I m etting the following error when jsp page is being compiled

weblogic.servlet.jsp.CompilationException: Failed to compile JSP /reports.jsp
reports.jsp:222:44: The deferred EL expression is not allowed since deferredSyntaxAllowedAsLiteral is false.

^_-----^

reports.jsp:222:44: The deferred EL expression is not allowed since deferredSyntaxAllowedAsLiteral is false.

^_-----^

August 2, 2011 at 7:56 AM

F. said...

Hi BalusC, I'm having the same problem reported by Victor Manuel.
I'm using JBoss AS 6 too.

Any suggestion?

August 3, 2011 at 12:23 PM

suraj chhetry said...

Hi,
At first, thank you for your word work. I have following above instruction and it is working file on glassfish but same things is not working on Jboss 6.1 . Form can't post. Does this work with Jboss 6.1 ???

September 19, 2011 at 9:01 AM

suraj chhetry said...

Hi,
At first, thank you for your word work. I have following above instruction and it is working file on glassfish but same things is not working on Jboss 6.1 . Form can't post. Does this work with Jboss 6.1 ???

September 19, 2011 at 9:01 AM

সেতু said...

Did any one find any solution so that
it works with JBoss 6.1 ?

October 8, 2011 at 9:55 AM

Martin Jondzic said...

This definitely doesn't work on JBoss 6.1.0 final, so BalusC, can you, please, provide additional hints? Thanx!

October 15, 2011 at 11:42 AM

Martin Jondzic said...

Hey guys,
an alternative (using Tomahawk library for JSF 2.0) provides solution for the file upload problem on JBoss 6.x.

Here is the link: <http://stackoverflow.com/questions/5418292/jsf-2-0-file-upload>.

This one was also written by BalusC.

October 15, 2011 at 9:57 PM

healeyb said...

Yep, t:inputFileUpload is the only solution I've been able to get working with container security. A significant limitation is that you can forget using ajax with this component, I've had to use full form submit, so maybe best to put it in it's own form. If you don't use maven it's a pain figuring out all the jars you need for tomahawk.

October 16, 2011 at 7:11 AM

John Page said...

Hi BaluC,
Many thanks for all the helpful code snippets and answers you have provided. I have been trying to get a fileUpload component with JSF going for the last 1 week without success. I first tried with PrimeFaces3.0M4 but the bean did not fire.
I also tried with Tomahawk 1.1.10 but am getting jar parsing errors in Tomcat.
Anyway, I am using MyFaces 2.0 with Tomcat 7.0.22 and would appreciate a solution. The example you have posted requires Mojarra and I am not keen to change my implementation.
Would appreciate any help.
Thanks,
John

November 17, 2011 at 9:39 AM

BalusC said...

Read [this answer](#). If it still fails, post a question over there with the exception in detail.

November 17, 2011 at 11:31 AM

John Page said...

Thanks for the example BaluC. I have decided to ditch MyFaces and move to Mojarra. I might also have to ditch Tomcat 7.0.22
(1) Mojarra 2.1.3 is not supported by Tomcat.
(2) Tomcat 7.0.22 does not support multi-part data parsing out of the box.
I also tried with Tomahawk but since it is not JSF 2.0 ready. I ran into some issues.
So my current architecture is
PrimeFaces 3.0M4 (file upload not firing)
Mojarra 2.1.3
Tomcat 7.0.22
Currently I am trying to implement your example which even after 3 years appears to be one of the few working examples of fileUpload in JSF.

November 17, 2011 at 11:41 PM

BalusC said...

Mojarra 2.1.3 works perfectly fine on Tomcat. 2.1.0 is the only Mojarra version which doesn't work on Tomcat. This was caused by a bug in the annotation scanner which accidently contained Glassfish-specific code. 2.1.1 and upwards works fine on Tomcat 7. I speak from own experience (my playground environment is always the latest Tomcat (7.0.22) + latest Mojarra (2.1.4)).

November 17, 2011 at 11:44 PM

BalusC said...

By the way, Tomahawk works fine for me on Mojarra 2.1.4 + Tomcat 7.0.22. You'd need to be more specific about the problem you're facing.

November 17, 2011 at 11:48 PM

SP said...

for JBoss6 support specify multipart config for jsf 2 servlet:

500000

December 20, 2011 at 6:28 AM

SP said...

"multipart-config" tag

December 20, 2011 at 6:32 AM

wxynxyo said...

Hi balusC:

i test your code but cannot find the java file "MultipartRequest". The java file exists in some jar or another?

Please guide me, i face the confuse thing for p:fileUpload and get the http error, i test all methods in web and cannot solution it,

December 23, 2011 at 2:41 AM

BalusC said...

Click the "You can find it all here" link in the 2nd paragraph.
December 23, 2011 at 8:58 AM
wxynxy said...
<i>This comment has been removed by the author.</i>
December 25, 2011 at 9:58 PM
wxynxy said...
Hi Balusc: use the p:fileUpload i face a confuse thing. When i start the project in my IDE, the file can be upload successfully when i click "Upload" button; But when i deploy the porject under ../tomcat/webapps/ and when click "Upload" button ,the Http Error appears; i remove the proxy, install the latest firefox,upload the latest Adobe flash, clean the cookie and add enctype="multipart/form-data" in , but the error still exists, would you mind guide me about it? my gmail is wxynxy@gmail.com, i'm very appreciate it.
December 25, 2011 at 10:01 PM
maniacneron said...
Hi Balusc; First of all thank you for all these valuable information and help. I have write my taglib in Eclipse Indigo added my tag and attributes. It works , but in xhtml eclipse does not auto complete my components and attriutes what should i do ? Is there a specific thing to do? i've tried same project with netbeans it works and auto completes without any extra effort. i have search a lot but could not find a clue. Thank you .
March 5, 2012 at 11:12 AM
Sabine said...
Thank youuuu so much for that ! I really don't know how I would have done it without your hepl ! It took me some time to make it works but by reading it several times and read the comments, finally it works ! yuuhuuu
March 6, 2012 at 8:14 PM
Big Allan said...
Thanks BalusC. Is there any way to get it to work with f:ajax ? When I use f:ajax on the upload button with the hh:inputFile tag set to execute I get a class cast exception in the decode method of FileRenderer (java.lang.ClassCastException: org.apache.catalina.connector.RequestFacade cannot be cast to jsf.file.upload.MultipartRequest) It works fine without the f:ajax tag. Thanks!
April 9, 2012 at 5:30 PM
pankajamu said...
Hi BalusC, May I know how can i find the .tld file for this custom file upload component?
May 3, 2012 at 10:16 AM
Pankaj said...
hi BalusC, May I know where is the Tag Class for this component??
May 3, 2012 at 11:18 AM
Brian said...
Thanks as usual BalusC for a great educational post. Saved me untold hours. Regarding the crashes people are seeing in FileRenderer#decode(), I came across the same problem tonight when otherwise everything else was working just fine (the filter was loading, etc.). It turns out my problem was that Netbeans has its own filter that wraps HttpServletRequestWrapper just as MultipartRequest does. The filter Netbeans is using is for its HTTP Monitor that is part of the integrated debugger/IDE (it's called MonitorRequestWrapper). I suspect there is something similar in Eclipse even though I'm not using it for my project so this could be causing crashes for others as well. My solution is to test to see if I can directly cast the request to MultipartRequest, and if not I try to see if it's wrapped by another filter. By doing this I managed to get things working. Here is the revised FileRenderer#decode() method: @Override public void decode(FacesContext context, UIComponent component) { rendererParamsNotNull(context, component); if (!shouldDecode(component)) { return; } String clientId = decodeBehaviors(context, component);

```
if (clientId == null) {
    clientId = component.getClientId(context);
}

Object request = context.getExternalContext().getRequest();

// Netbeans MonitorRequestWrapper owns the MultipartRequest
// during debugging, so dig a bit to find the real request.
if ( ! ( request instanceof MultipartRequest ) ) {
    if ( request instanceof HttpServletRequestWrapper ) {
        request = ((HttpServletRequestWrapper) request).getRequest();
    }
}
File file = request instanceof MultipartRequest ?
((MultipartRequest) request).getFile(clientId) : null;

// If no file is specified, set empty String to trigger validators.
((UInput) component).setSubmittedValue((file != null) ? file : EMPTY_STRING);
}
```

Hope it helps,

par

May 9, 2012 at 5:02 AM

Primal Intellect said...

Thank you very much for all of your work and tutorials on this balusc!!

I have a similar issue as Alex. Is there a way to restrict the maximum file size to prevent large files from being uploaded?

Right now if a file is larger than MAX_FILE_SIZE in FileValidator it still gets uploaded which seems really bad from a performance standpoint. Is there a way to prevent this?

TIA!

May 18, 2012 at 8:25 PM

noboundaries said...

Thank you and I see no file upload button but the below warning in browser. May be the minor mistake but don't know how to solve. Please suggest.

Warning: This page calls for XML namespace http://balusc.net/jsf/html declared with prefix hh but no taglibrary exists for that namespace.

June 15, 2012 at 7:04 AM

noboundaries said...

Sorry, I wrongly specified the taglib xml name in web.xml. Now it works!

June 15, 2012 at 7:28 AM

Бранко Илић said...

Thank you, BalusC! In Tomcat 7.0.27 Context attribute allowCasualMultipartParsing still has be set on "true" :)

June 29, 2012 at 5:54 PM

Fahim Parkar said...

I have same error as kedves_uram said...

"Expression Error: Named Object: HtmlInputFile not found.

javax.faces.FacesException: Expression Error: Named Object: HtmlInputFile not found.

July 3, 2012 at 6:33 AM

Pedram said...

thank you very much!

August 31, 2012 at 10:18 AM

Alex Mickel said...

Hello. I am using Eclipse Indigo with JSF 2.0 and Servlet 3.0. The Glassvish is V3.1.1. and Mojarra 2.1.3. I don't get it to work. I always get an exception when starting:

HTTP status 404 -
type Status report

message
descriptionThe requested resource () is not available.

Does anyone know what this is? I get the same when I try the File Upload with MyFaces, PrimeFaces, Tomahawk etc.

It would be great if someone knows an answer to this.

September 1, 2012 at 9:37 AM

Chaman k lalwani said...

Thanks BalusC for posting such a nice tutorial. i just want to ask a simple question that, Is TagHandler class necessary for creating custom component in JSF 2.0?

September 27, 2012 at 4:46 AM

Chaman k lalwani said...

I followed the steps mentioned in this link <http://docs.oracle.com/javaee/6/tutorial/doc/bnawa.html>, but i found that my setProperties and other setter methods never called in Taghandler class and my tag class extends UIComponentELTag. so please help me in creating custom component.

September 27, 2012 at 4:51 AM

Netto said...

This comment has been removed by the author.

October 9, 2012 at 1:50 PM

Netto said...

Hello BalusC,

I did all, but I have a problem running the app:

INFO: 2012-10-09 12:01:54,645 [JSF-Tester] DEBUG ReducedHTMLParser:425 - DOCTYPE found at line 1

INFO: Instantiated an instance of org.hibernate.validator.engine.resolver.JPATraversableResolver.

INFO: Text: ddd

INFO: File: null

INFO: Check: [check1]

INFO: 2012-10-09 12:02:27,558 [JSF-Tester] DEBUG ReducedHTMLParser:425 - DOCTYPE found at line 1

you Know what could be The problem??

October 9, 2012 at 1:52 PM

ibrahim demir said...

Hi BalusC,

this example is so good but not reusable in other projects. I googled a lot to develop a JSF2 component library. But i couldn't find enough information. Do you know how to make this example to external component library(jar file).

October 17, 2012 at 2:06 AM

Sven Tschui said...

Jboss AS 7 needs in web.xml:

```
<servlet>
<servlet-name>Faces Servlet</servlet-name>
<servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
<load-on-startup>1</load-on-startup>
<multipart-config/>
</servlet>
```

December 4, 2012 at 6:21 AM

Jhonatan Forero said...

Hi BalusC!, I need to implement in the upload file when client select a file a PGP encryption from client side with JSF 2 (Create a faces component) or primefaces 2.2 (I dont see anything from client side), because the file has sensitive information, encryption from the server does not help, can you give me some help or any idea?

Thanks Balusc!

February 22, 2013 at 10:52 PM

Jhonatan Forero said...

<http://stackoverflow.com/questions/15041487/pgp-encryption-from-client-side-js-with-jsf-2>

February 23, 2013 at 10:34 AM

[Post a Comment](#)

[Newer Post](#)

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)

[Add by Google](#)

[JSF Examples](#)

[NetBeans JSF](#)

[JSF UI Components](#)

[Faceset JSF](#)