Generalitat de Catalunya
Centre de Telecomunicacions
i Tecnologies de la Informació

Framework Corporatiu J2EE

Servei de Tags de Presentació

Versió 1.2

Barcelona, 13 / febrer / 2007

Històric de modificacions

Data	Autor	Comentaris	Versió
09/12/2005	Atos Origin, sae openTrends	Versió inicial del document	1.0
13/02/2007	Atos Origin, SAE	Versió 1.2 d'OpenFrame	1.2

Llegenda de Marcadors

Índex

1. Introducció

1.1. Propósit

El Servei de Tags té com a propósit principal oferir un ventall de components a utilitzar dins de les nostres pàgines. En l'actualitat es basen en l'ús de pàgines JSP (Java Server Pages), per ser ara mateix la tecnologia de presentació més extesa.

openFrame, en el seu afany de proporcionar la màxima flexibilitat i facilitat als desenvolupadors ha seleccionat, extés i millorat algunes propostes de la comunitat. En molts dels components es fan servir tecnologies Web d'última generació i s'extenen solucions openSource ja existents. Tammateix s'ha definit una política de configuració que facilita l'herència de valors de les propietats entre components, aspecte no assolible amb la tecnologia de tags JSP.

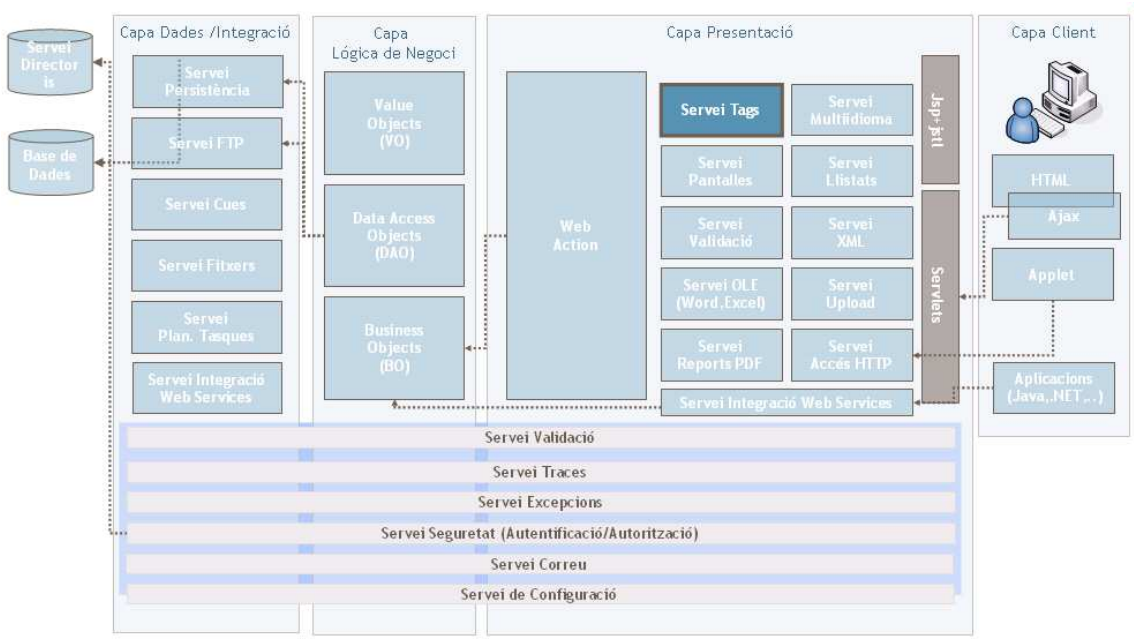
En l'actualitat existeix una nova generació de desenvolupament Web anomenada AJAX [1](**Asynchronous JavaScript And XML)**. Tot i que el desenvolupador no necessita del coneixement d'aquest tecnologia, openFrame s'ha adaptat a aquesta nova generació en la implementació interna dels seus tags.

Es considera prerequisit per a la lectura d'aquest document el coneixement en HTML, CSS i l'ús de Struts.

1.2. Context i Escenaris d'Ús

El Servei de Tags es troba dins dels serveis de Presentació de openFrame.

1 Podeu veure una introducció a 'http://www.adaptivepath.com/publications/essays/archives/000385.php'



Per a la utilització dels tags de openFrame es requereix d'un entorn basat en les següents tecnologies:

- JSP
- JSTL
- Struts

1.3. Versions i Dependències

En el present apartat es mostren quines són les versions i dependències necessàries per fer ús del Servei.

1.3.1. Versions

No s'han produït canvis respecte la versió 1.0.

1.3.2. Dependències Bàsiques

Nom	Tipus	Versió	Descripció
ajaxtags	jar	1.1	
dwr	jar	1.1-beta1	
jsp-api	jar	2.0	http://java.sun.com/products/jsp
jstl	jar	1.1.1	
nekohtml	jar	0.8.1	

Nom	Tipus	Versió	Descripció
openFrame-core	jar	1.0	
openFrame-services-configuration	jar	1.0	
openFrame-services-exceptions	jar	1.0	
openFrame-services-i18n	jar	1.0	
openFrame-services-logging	jar	1.0	
openFrame-services-validation	jar	1.0	
openFrame-services-web	jar	1.0	
oro	jar	2.0.8	http://jakarta.apache.org/oro/
js	jar	1.5R4.1	
servlet-api	jar	2.4	
spring	jar	1.2.5	http://www.springframework.org
struts-layout	jar	1.2_20051027	
struts-menu	jar	2.3	
struts	jar	1.2.7	

Per cada servei openFrame fer ús també de les dependències requerides per ell.

Es recomana consultar la referència del servidor d'aplicacions en el que es farà el deployment de l'aplicació per conéixer si les versions aquí indicades estan permeses.

1.4. A qui va dirigit

Aquest document va dirigit als següents perfils:

- Programador. Per configurar els tags i editar les pàgines
- Dissenyador. Per definir els estils i la composició de les pàgines

1.5. Documents i Fonts de Referència

Ubicació dels labels	http://illuminosity.net/thoughts/archives/styling_forms/
Llibreria per accessKey	http://www.gerv.net/software/aul/

Llibreria Struts Layout	http://struts.application-servers.com/
Formularis accessibles	http://www.websemantics.co.uk/tutorials/accessible_forms/#texttop
Advanced CSS	http://www.yourhtmlsource.com/stylesheets/advancedcss.html
Overlib	http://www.bosrup.com/web/overlib/
Dom tooltip	http://www.mojavelinux.com/projects/domtooltip/HOWTO.html
Dynarch Calendar	http://www.dynarch.com/projects/calendar/
DHTML Internet Explorer	http://msdn.microsoft.com/workshop/author/dhtml/reference/objects.asp
Prototype library	http://prototype.conio.net/
Value List	http://valuelist.sourceforge.net/
Struts Layout	http://struts.application-servers.com/doc/tags/layout.html
Ditchnet JSP Tags	http://209.61.157.8:8080/taglibs/

1.6. Glossari

TLD (Tag Library Descriptor)

Un TLD és un fitxer en format XML que permet a un desenvolupador definir un conjunt de tags. Per cada tag es poden descriure els seus atributs i la classe associada que generarà el contingut de sortida a la pàgina en la que es faci servir.

Tot i que l'ús de tags específics permet eliminar l'ús de codi Java des de les nostres pàgines JSP, no eliminen la necessitat de crear codi Java. S'han de crear igualment classes Java que facin el treball del tag. Aquestes classes han de extendre les classes definides al package 'tagext' de J2EE.

2. Descripció Detallada

2.1. Arquitectura i Components

L'arquitectura del Servei de Tags és basa en l'ús de llibreries de tags JSP.

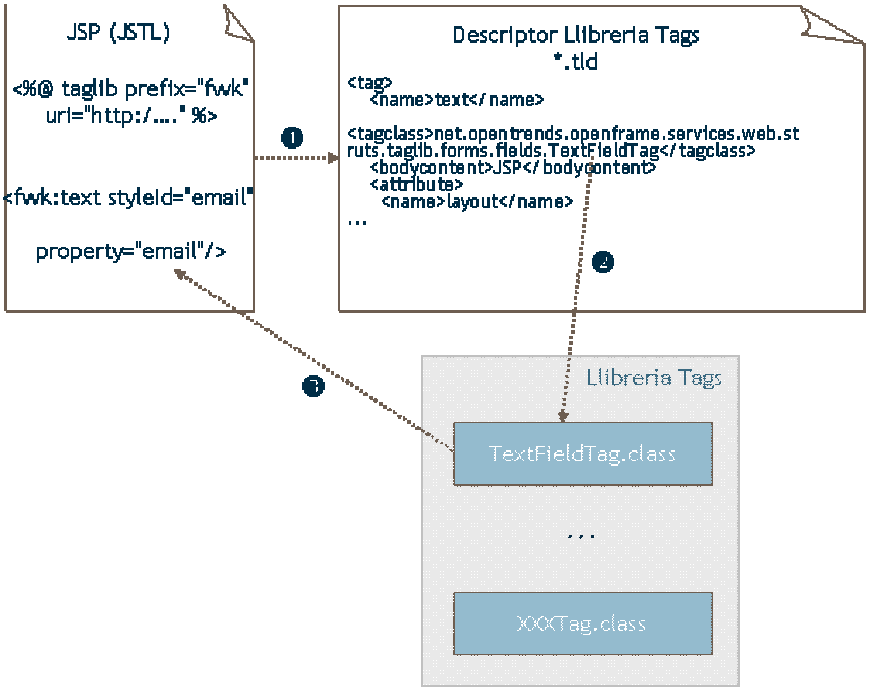
Es recomana una lectura prèvia de la url 'http://java.sun.com/products/jsp/tutorial/TagLibrariesTOC.html' per conéixer què és una llibreria de tags, cóm es creen i cóm s'utilitzen.

A mode de resum, l'arquitectura que hi ha al darrera de les llibreries de tags és la mostrada a continuació:

1) El contenidor del Servidor d'Aplicacions en generar el contingut de sortida a partir de la pàgina JSP troba la referència '<%taglib>' on s'indica la localització de la llibreria de tags que resol tot tag de la pàgina que contingui el prefix 'fwk'.

2) Quan troba el tag <fwk:text> accedeix al descriptor tld i troba quina és la classe que generarà el contingut a substituir en la localització del tag

3) El tag realitza la sortida a partir dels atributs passats a la pàgina JSP. Aquest contingut s'incorpora a la pàgina



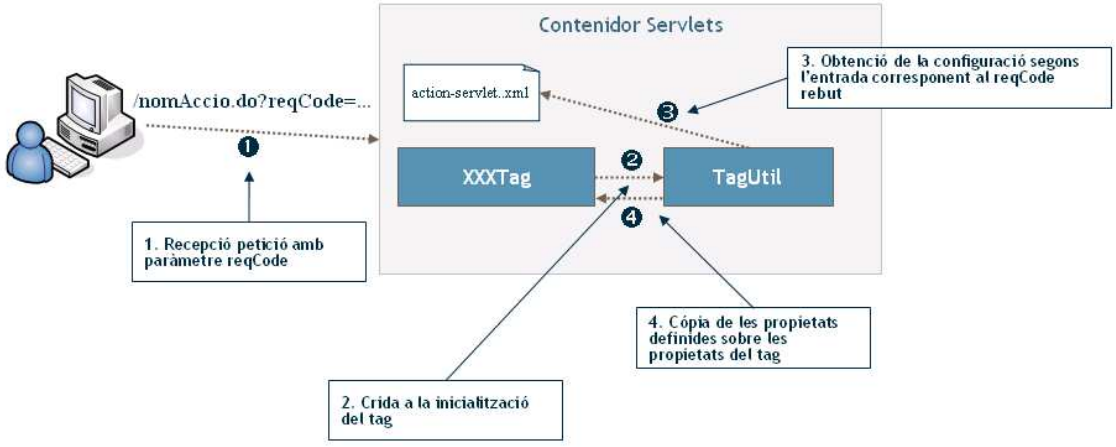
Seguint la filosofia d'independència màxima de la implementació openFrame defineix interfícies per tots els tags. En l'actualitat s'ofereixen implementacions basades en JSP i Struts. Degut a la filosofia que permet configurar de forma externa els tags de les pàgines, la migració a altra tecnologia serà molt més fàcil.

2.1.1. Arquitectura de Procediment d'Inicialització Intern

La configuració dels tags es realitza mitjançant el Servei de Configuració com si de classes planes es tractessin.

En el model inherent a JSP, és el contenidor qui instancia els tags. Això implica que no ens permet de fer ús directe del patró 'Dependency Injection'. openFrame ha extés aquest comportament per tal de poder inicialitzar els tags i les seves propietats mitjançant el Servei de Configuració. Aquest procediment es segueix per tots els tags de formulari, mentre que es deixa de banda aquest mecanisme per tags específics de layout i altres tags d'utilitat.

El procediment general seguit per a la inicialització d'un tag és el següent:



1) Es rep la petició de l'usuari (segons per exemple un submit) en el que arriba un paràmetre 'reqCode' que indica el mètode de l'acció Struts que s'executarà. Una vegada finalitzada l'acció aquesta retorna la pàgina JSP que mostrarà el contingut de la resposta.

2) En la generació del contingut de sortida el contenidor del servidor crida a la inicialització de cadascun dels tags inclosos a la pàgina JSP.

   Imaginem que a la pàgina JSP tenim un component definit de la següent forma:

```
<fwk:text styleId="firstname" property="firstname"/>
```

És obligatori que definim aquestes 2 propietats a les nostres pàgines. L'atribut 'property' és equivalent al que fa servir Struts, que ens permet lligar quin atribut de l'objecte volem presentar. L'atribut 'styleId' servirà per obtenir la configuració del tag des del fitxer de configuració.

El propi tag, dins el seu mètode d'inicialització crida a la classe 'TagUtil' indicant-li quin és el seu 'styleId' per obtenir les seves propietats.

3) La classe TagUtil consulta el fitxer de configuració 'action-servlet.xml' i obté el bloc definit dins l'entrada corresponent al 'reqCode' que ha arribat com a paràmetre i les propietats definides per el 'styleId' del tag.

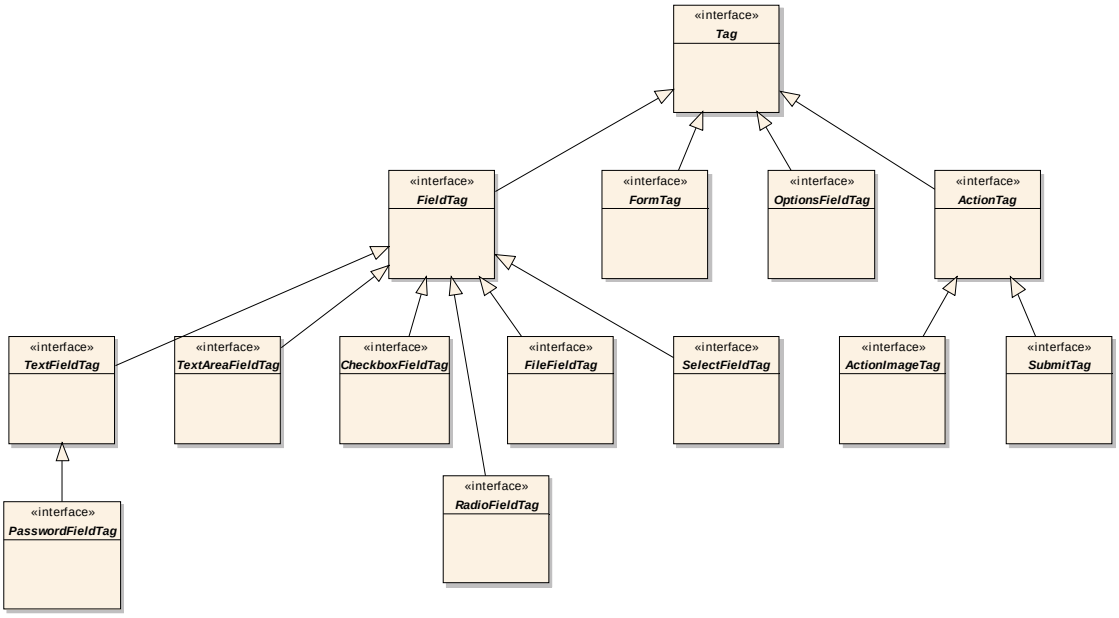
```
<property name="tagsConfiguration">
        <map>
                <entry key="*" ❶>
                        <list>
                        ...
                                <bean id="..." class="...">
                                        <property name="styleId"
                                value="nombreEstiloId" ❷/>

                                </bean>
                        </list>
                </entry>


                <entry key="reqCode">
                        <list>
                                <bean ...>
                                        <property name="styleId"
                                value="nombreEstiloId"/>

                                </bean>
                        </list>
                </entry>
                </map>
```

❶ Segons el reqCode que arribi com a paràmetre a la url '...?reqCode=...' s'escollirà una entry o una altra per configurar els tags de la pàgina.
❷ L'estil indicat lliga amb l'atribut 'styleId' que s'ha definit a la pàgina JSP

4) El tag, mitjançant commons beanutils copia les propietats definides sobre les seves propietats. A continuació realitza la generació de la sortida segons les propietats definides.

2.1.2. Interfícies i Components Genérics

Els tags del servei es troben definits mitjançant interfícies que defineixen amb mètodes què s'espera de les implementacions.



En el gràfic mostrat a dalt es pot veure la jerarquia d'interfícies definida per openFrame.

Component	Package	Descripció
Tag	net.opentrends.openframe.services .web.taglib	Interfície base de tots els tags. Defineix principalment la necessitat que totes les implementacions hagin definit mètodes d'accés als atributs: • 'styleId'. Identificador del tag dins la pantalla • 'pageContext'. Obtenció del context de la pàgina • validationService. Referència al servei de validació • i18nService. Referència al servei d'internacionalització
FieldTag	net.opentrends.openframe.services .web.taglib	Interfície de definició de tots els tags que permetin l'entrada de dades en un formulari
FormTag	net.opentrends.openframe.services .web.taglib	Interfície de definició d'un tag que representa un formulari
OptionsFieldTag	net.opentrends.openframe.services .web.taglib	Interfície de definició d'un tag que representi els valors d'una llista d'opcions
ActionTag	net.opentrends.openframe.services .web.taglib	Interfície tags d'acció d'un formulari
TextFieldTag	net.opentrends.openframe.services .web.taglib	Interfície tag entrada d'una dada d'una sola línia en un formulari

Component	Package	Descripció
TextAreaFieldTag	net.opentrends.openframe.services.web.taglib	Interfície tag entrada de dada de vàries línies en un formulari
CheckboxFieldTag	net.opentrends.openframe.services.web.taglib	Interfície tag de marcar un valor en un formulari
FileFieldTag	net.opentrends.openframe.services.web.taglib	Interfície tag per enviar un fitxer dins un formulari
SelectFieldTag	net.opentrends.openframe.services.web.taglib	Interfície tag de selecció d'una llista de valors
ActionImageTag	net.opentrends.openframe.services.web.taglib	Interfície tag per usar una imatge de botó per acceptar un formulari i enviar-lo
SubmitTag	net.opentrends.openframe.services.web.taglib	Interfície tag per usar una botó per acceptar un formulari i enviar-lo
PasswordFieldTag	net.opentrends.openframe.services.web.taglib	Interfície tag entrada de dades que no es visualitzen per pantalla
RadioFieldTag	net.opentrends.openframe.services.web.taglib	Interfície tag per selecció d'un radio button

2.1.3. Implementació basada en Struts

La implementació basada en Struts de les interfícies mostrades es troben dins el package 'net.opentrends.openframe.services.web.struts.taglib.forms.fields'. Els seus noms coincideixen amb els de les interfícies.

Els tags actuals de openFrame es basen en extensions dels tags de Struts-Layout. Per evitar una màxima dependència amb aquesta implementació s'han definit interfícies de tots els tags (tal i com s'ha vist en l'anterior apartat). Veure l'apartat 'Configuració i Instal·lació' per a més detalls.

2.2. Instal·lació i Configuració

Prerequisit: Per al correcte funcionament d'aquest servei és important que prèviament s'hagin realitzat les configuracions especificades a l'apartat 'Configuració Bàsica' del document 'Serveis de Presentació'.

2.2.1. Instal·lació

Per a la instal·lació del Servei de Tags es requereix de l'ús del fitxer 'openFrame-services-web.jar' i les dependències indicades a l'apartat 'Versions i Dependències'.

2.2.2. Configuració General

A diferència del tradicional mecanisme de tags JSP en el que dins la pàgina s'especifiquen les propietats del tag, openFrame va més enllà i permet tractar els tags com classes Java normals, a fi que es puguin injectar les seves propietats mitjançant el Servei de Configuració.

Aquesta aplicació es fa només pels components d'entrada de formularis, degut a la necessitat que tenen diferenciada segons l'acció realitzada.

openFrame fa ús del Servei de Configuració per a definir els diferents comportaments d'una mateixa plana de presentació segons el cas d'ús. Això implica que una mateixa plana JSP pot tenir diferents configuracions de visualització i comportament dinàmic, descrites de manera declarativa. Les nostres pàgines JSP seran molt simples, ja que s'externalitzaran les seves propietats en aquests fitxers de configuració, en els quals podem utilitzar la potència de l'herència per definir propietats comuns.

Per cada acció Struts configurada al fitxer 'action-servlet.xml' definirem la propietat 'tagsConfiguration' on definirem el següent format:

La configuració del Servei de Tags implica els següents pasos:

1) Definir la propietat de configuració de l'acció
2) Definir les propietats de cada tag

Definició de la configuració per l'acció

```
<bean name="/action"
class="....XXXAction">
       ...
       <property name="tagsConfiguration">
              <map>
                     <entry key="*">
                            <list>
```

La definició de les propietats que s'usaran pels tags d'una pantalla es poden definir per cada acció de l'aplicació. Així, per exemple podríem representar en una creació alguns aspectes de forma diferent.
Per definir quines són les propietats dels tags per un reqCode en concret es definirà la propietat 'tagsConfiguration'. Aquesta propietat és un mapa, on:

- La clau es farà servir com a cerca de la configuració segons el 'reqCode' rebut al paràmetre.

  Si es defineix amb el valor '*' es considerà com la configuració per defecte, en tant que s'aplicarà en els següents casos:

  a) No arriba 'reqCode' a la url
  b) Si no es troba configuració d'un determinat tag en la configuració segons el 'reqCode' rebut, es cercarà en la configuració per defecte '*'
  c) Si la definició específica hereta de la definició realitzada a la configuració per defecte '*'

- El valor de l'entrada correspondrà a una llista de beans que definirà les propietats de cadascun dels tags

Exemple:

```
<bean name="/accounts❶"
class="net.opentrends.openframe.samples.jpetstore.struts.action.AccountAction">
   ...
   <property name="tagsConfiguration❷">
          <map>
                 <entry key="*"❸>
                        <list>

                               <bean id="idForm" ...>
                                      <property name="styleId"
                                      value="actionForm"/>
                               </bean>


                               <bean id="idFirstName" ...>
                                      <property name="styleId"
                                      value="firstname"/>
                                      <property name="accesskey"
   value="forms.accountForm.field.firstname.accesskey"/>

                                      <property name="key"
   value="forms.accountForm.field.firstname"/>                        ...

                               </bean>


                 <entry key="edit"❹>
                        <list>
                               <bean id="idFirstName" parent="textFieldTag">
                                      <property name="styleId"
                               value="name"/>

                               </bean>
                        </list>
                 </entry>
                 </map>
          </property>
   </bean>
```

❶ Acció de Struts amb injecció de les seves propietats (veure document 'Serveis de Presentació' per a més referència).
❷ Definició de la propietat 'tagsConfiguration'
❸ Per qualsevol reqCode es farà servir la configuració dels tags aquí indicada
❹ En cas que arribés el paràmetre 'reqCode' amb valor 'edit' es farà servir aquesta configuració

Una vegada definida la propietat de configuració de l'acció podem definir les propietats de cadascun dels tags.

Definició de les propietats d'un tag

```
<bean name="/action"
class="....XXXAction">
       ...
       <property name="tagsConfiguration">
              <map>
                     <entry key="*">
                            <list>
                                   <bean id="" class="">
                                          ...
                                   </bean>
```

La configuració d'un tag és equivalent a la configuració de qualsevol altre element (consultar el document 'Servei de Configuració' per més referència).

A l'atribut 'class' doncs s'especificarà quin és la classe concreta d'implementació del tag que farem servir.

En el cas de Struts farem ús de les següents classes:

- net.opentrends.openframe.services.web.struts.taglib.forms.fields.FormTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.OptionsFieldTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.ActionTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.TextFieldTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.TextAreaFieldTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.CheckboxFieldTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.FileFieldTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.SelectFieldTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.ActionImageTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.SubmitTag
- net.opentrends.openframe.services.web.struts.taglib.forms.fields.PasswordFieldTag

En tots ells és necessari configurar les següents propietats:

Propietat	Requerit	Descripció
styleId	Sí	Identificador del tag. És l'identificador de connexió amb el camp 'styleId' del tag a la pàgina JSP.

Adicionalment, depenent del tag podrem configurar altres propietats. Tots ells permeten configurar les propietats dels tags en el que es basen. Podem per tant configurar qualsevol dels atributs de Struts mitjançant el fitxer de configuració.

La potència real d'aquesta forma de treballar versus els clàssics tags JSP és que podem usar l'herència i definir propietats genériques a un grup de tags. Així, podem definir propietats comunes a varis tags mitjançant el Servei de Configuració tal i com es mostra en el següent exemple:

```
<!-- TAG's default configuration -->
   <bean id="formTag"
   class="net.opentrends.openframe.services.web.struts.taglib.forms.FormTag"/>

   <bean id="actionImageTag"
class="net.opentrends.openframe.services.web.struts.taglib.forms.ActionImageTag"/>
   <bean id="submitTag"
class="net.opentrends.openframe.services.web.struts.taglib.forms.SubmitTag"/>

   <bean id="textFieldTag"
   class="net.opentrends.openframe.services.web.struts.taglib.forms.fields.TextFie
ldTag">
           <property name="i18nService" ref="i18nService"/>
           <property name="validationService" ref="webValidationService"/>
           <property name="layout" value="true"/>
   </bean>
   ...

   <bean id="checkboxFieldTag"
class="net.opentrends.openframe.services.web.struts.taglib.forms.fields.CheckboxFi
eldTag">
           <property name="i18nService" ref="i18nService"/>
           <property name="validationService" ref="webValidationService"/>
           <property name="layout" value="true"/>
   </bean>
```

En aquest exemple, es defineixen les propietats comunes als diferents tipus de components. Així podem fer ús de l'herència per definir els components específics de la nostra aplicació com per exemple:

```
<bean parent="textFieldTag">
   <property name="styleId" value="id"/>
   <property name="mode" value="E,I,I"/>
   <property name="key" value="forms.accountForm.field.id"/>

</bean>
```

Aquí s'ha eliminat la necessitat de definir el id del tag i s'ha fet ús de l'herència mitjançant el tag 'textFieldTag' que fa referència al '<bean id=”textFieldTag”>' que s'ha definit en el fitxer extern.

Podem, per últim, estructurar la següent forma el nostre fitxer de l'aplicació:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
   <!-- TAG's default configuration -->
```

```
    <bean id="formTag"
    <bean id="fieldTag">
            ...
    </bean>

    <bean id="textFieldTag"
    class="net.opentrends.openframe.services.web.taglib.config.TextFieldTagConfigur
ation" parent="fieldTag">
    </bean>


    <import resource="action-servlet-accounts.xml"/>
    <import resource="action-servlet-categories.xml"/>
    <import resource="action-servlet-products.xml"/>
    <import resource="action-servlet-items.xml"/>

</beans>
```

On mitjançant el tag 'import' es fa referència a diferents fitxers de configuració (grups per acció, per una sola acció, etc.)

A continuació s'ofereix explicació de la configuració de cadascun dels tags.

2.2.3. Configuració del Tag 'ConfigurationTag'

Ús: Obligatori, doncs genera les llibreries dels components

El tag Configuration de openFrame s'encarrega de generar totes les llibreries necessàries que requereixen els tags.

Per a generar les llibreries necessàries accedeix al fitxer de propietats de Spring i per cada tag definit obté les seves propietats. Segons els valors d'aquestes propietats genera les llibreries.Per exemple, en el cas de especificar 'showCalendar' a un TextField, generarà les llibreries de calendari, així com la llibreria de literals de calendari de l'idioma de l'usuari entre d'altres.

Atributs:

Atributs	Requerit	Descripció
id	Sí	Usar el valor 'configurationTag'
class	Sí	Usar 'net.opentrends.openframe.services.web.struts.taglib.configuration.ConfigurationTag'

Propietats:

Atributs	Requerit	Descripció
styleId	Sí	Usar el valor 'defaultConfiguration'

Atributs	Requerit	Descripció
i18nService	Sí	Referència al Servei d'Internacionalització

Exemple:

```
<bean id="configurationTag"
class="net.opentrends.openframe.services.web.struts.taglib.configuration.Configura
tionTag">
          <property name="styleId" value="defaultConfiguration"/>
          <property name="i18nService" ref="i18nService"/>
</bean>
```

I a la pàgina JSP (es recomana definir-lo al template de Struts Layout):

```
<fwk:configuration styleId="defaultConfiguration" />
```

2.2.4. Configuració dels tags de Formulari

Configuració del Tag Form

Aquest tag es basa en Struts, pel que sempre podem definir les propietats que el tag 'form' de Struts proporciona. En aquest apartat només es definiran aquelles propietats adicionals que ofereix openFrame.

Atributs adicionals:

Atribut	Requerit	Valor
reqCode	Sí	Aquest paràmetre permet especificar quin nom de funció de la Action es cridarà en el moment de fer el submit del formulari. Exemple: Si especifiquem com a 'reqCode=save' es cridarà al mètode 'save' de l'acció de Struts.
Layout	Sí	Aquesta propietat permet definir si volem generar o no el layout pel formulari. Per defecte, el tag crea un <table> que agruparà els components input interns. Aquests components interns generaran '<tr>'<td>' de forma automàtica a menys que configurem la seva propietat 'layout' a false. En qualsevol cas, també s'ofereixen tags que ens permetran definir el nombre de columnes i files, crear una fila on ubicar tots els components (cas en el que no crearan <tr>), etc. (veure 'Tags de Layout').
validationProperties	No	Mitjançant aquesta propietat de tipus 'Properties' podem

Atribut	Requerit	Valor
		especificar com es farà la validació de les dades entrades al formulari. Veure informació detallada del seu ús a continuació

- Configuració de l'atribut 'validationProperties'

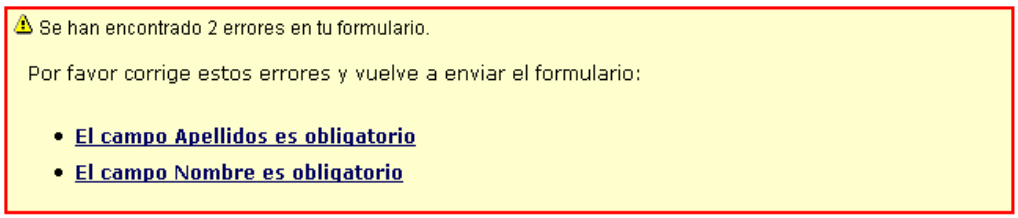
```
<property
name="validationProperties">
      <props>...
</property>
```

Dins el tag <props> podem definir les següents propietats:

Atribut	Requerit	Valor
validationType	Sí	Es permeten els valors: • SERVER. La validació es realitzarà mitjançant Ajax contra el Servei de Validació en el servidor. El resultat de la validació apareixerà a la mateixa pàgina HTML del client • CLIENT. La validació es realitza mitjançant Javascript generat al client
validatorName	Depén validationnType	Especifica quin és el nom de validador definit al fitxer 'validation.xml' del servidor que es farà servir per validar els camps entrats (veure 'Servei de Validació' per més referència) En el cas que s'hagi usat 'SERVER' com a 'validatorType', no cal indicar 'validatorName' si es vol usar com a nom de validador el 'pojoClass' configurat a l'acció associada al formulari.

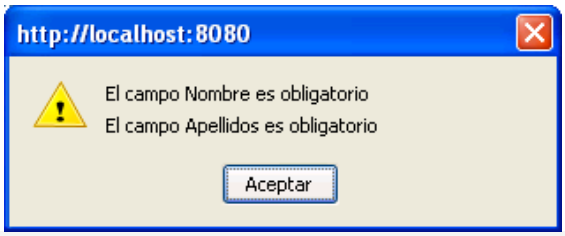
Exemple segons validationType 'SERVER':

```
<property name="validationProperties">
   <props>
          <prop key="validationType">SERVER</prop>
   </props>
</property>
```



Exemple segons validationType 'CLIENT':

```
<bean parent="formTag">
          <property name="styleId" value="actionForm"/>
   <property name="validationProperties">
          <props>
                 <prop key="validationType">CLIENT</prop>
                 <prop key="validatorName">account</prop>
          </props>
```




```
          ...
</bean>
```

Configuració del Tag Submit

Aquest tag es basa en Struts, pel que sempre podem definir les propietats que el tag 'submit' de Struts proporciona.

En aquest apartat només es definiran aquelles propietats adicionals que ofereix openFrame. Cal tenir en compte que – seguint el patró de configuració establert pels tags de formulari - podem configurar en el fitxer 'xml' totes les propietats, deixant la pàgina JSP força senzilla.

```
<fwk:submit styleId="saveActionImage"/>
```

En el fitxer de configuració xml podem definir qualsevol de les propietats del tag de Struts i adicionalment:

Atribut	Requerit	Valor
styleId	Sí	Usar el valor 'styleId' definit a l'atribut del tag a la pàgina JSP
reqCode	Sí	Valor que indicarà quin reqCode s'enviarà com a paràmetre. Aquest valor servirà per localitzar quin es el mètode que s'executarà a l'Action.
Mode	Sí	Aquesta propietat permet especificar cóm ha de visualitzar-se el botó segons el mode del formulari. S'ha d'usar el següent format: X,Y,Z

Atribut	Requerit	Valor
		On X,Y i Z poden prendre els següents valors: • D: Indica que el botó es mostrarà (display) • N: Indica que el botó no es mostrarà (not displayed) • F: Indica que el botó es mostrarà desactivat (frozen) Cadascuna de les posicions (X, Y, Z) correspon a cóm es visualitzarà el botó segons si el mode del formulari és de creació, d'edició o de consulta respectívament. Exemple: mode="N,D,D"
Key	Sí	Clau del fitxer de propietats de multidioma per generar el text del botó

Configuració del Tag ActionImage

Aquest tag extén el tag 'image' de Struts, pel que sempre podem definir les propietats que el tag de Struts proporciona (srcKey per referència a la clau que defineix el path de la imatge, src per la url de la imatge, etc.).

En aquest apartat només es definiran aquelles propietats adicionals que ofereix openFrame. Cal tenir en compte que – seguint el patró de configuració establert pels tags de formulari - podem configurar en el fitxer 'xml' totes les propietats, deixant la pàgina JSP força senzilla.

```
<fwk:actionImage styleId="saveActionImage"/>
```

En el fitxer de configuració xml podem definir qualsevol de les propietats del tag de Struts i adicionalment:

Atribut	Requerit	Valor
styleId	Sí	Usar el valor 'styleId' definit a l'atribut del tag a la pàgina JSP
reqCode	Sí	Valor que indicarà quin reqCode s'enviarà com a paràmetre. Aquest valor servirà per localitzar quin es el mètode que s'executarà a l'Action.
Mode	Sí	Aquesta propietat permet especificar cóm ha de visualitzar-se el botó segons el mode del formulari. S'ha d'usar el següent format: X,Y,Z On X,Y i Z poden prendre els següents valors:

Atribut	Requerit	Valor
		• D: Indica que el botó es mostrarà (display) • N: Indica que el botó no es mostrarà (not displayed) • F: Indica que el botó es mostrarà desactivat (frozen) Cadascuna de les posicions (X, Y, Z) correspon a cóm es visualitzarà el botó segons si el mode del formulari és de creació, d'edició o de consulta respectívament. Exemple: mode="N,D,D"

Configuració de Propietats comunes a tots els tags d'entrada de dades (inputs) de formulari

Quan desenvolupem una aplicació en la que intervenen formularis que es poden mostrar de manera diferenciada segons els privilegis de l'usuari o l'acció escollida existeixen 2 formes tradicionals de definir-los:

1) Creant una pàgina JSP per cada possible presentació del mateix formulari
2) Imbricant codi condicional que faci el control (segons els privilegis, el paràmetre de la url, etc.) i executi la creació dels tags de diferents maneres

Tant una com l'altra solució requereixen d'un esforç massa gran. En la primera opció, un canvi que impliqui afegir un camp o retocar un ja existent pot requerir de la necessitat de modificar vàries pàgines JSP, mentre que en la segona solució la pàgina es fa molt difícil de llegir i mantenir.

openFrame permet l'ús de modes de formulari que s'aplicaran als components continguts en ell. Segons el mode especificat, els components es mostraran d'una o un altra forma (ocults, editables, no editables, etc.)

Els tags input de openFrame es basen en els tags proporcionats per Struts, de forma que podem sempre definir les seves propietats als nostres fitxers de configuració.

En aquest apartat només es mostren les propietats adicionals que ofereix openFrame:

Propietat	Requerit	Valor
key	Sí	Clau del literal a generar com a 'label' del component.
layout	No	Indicar 'false' si no es volen generar els tags '<tr>','<td>' per als components. Per defecte és 'true' si no indiquem aquesta propietat
accessKey	No	Permet fer ús d'accés per teclat al component. S'ha d'especificar la clau del fitxer d'internacionalització que defineix el caràcter que ha de subratllar (del valor del 'key' definit) i considerar per generar l'event de focus si es fa servir 'ALT+valor key'

Propietat	Requerit	Valor
mode	Sí	Mode de visualització del component segons el mode que s'hagi assignat al formulari (veure apartat 'Utilització del Servei' per conéixer cóm assignar el mode del formulari). S'ha d'indicar un valor amb 3 parts separades per ','. La primera indica en quin mode apareixerà el component si el mode del formulari és de creació, la segona part en quin mode ha d'aparéixer si el mode del formulari és d'edició i la tercera part en quin mode ha d'aparéixer si el mode del formulari és de consulta. Exemple: 'E,E,I' A continuació s'ofereix més detall dels valors possibles i del seu ús.
required	No	Mostrar el camp com obligatori. No cal definir-lo si s'ha definit al 'Servei de Validació' que el camp és obligatori. A continuació s'oferirà més detall dels valors possibles i del seu ús.
tooltipKey, tooltipTitleKey i tooltipOptions	No	Permeten generar un tooltip d'ajuda al component A continuació s'oferirà més detall dels valors possibles i del seu ús.
selectOnFocus	No	En rebre el focus el component tindrà tot el seu contingut seleccionat. Especificar 'false' si no es vol utilitar aquest comportament per defecte.
i18nService	Sí	Referència al servei d'internacionalització
validationService	Sí	Referència al servei de validació

- **'key' i 'layout '(Generació de etiquetes i layout als components)**

En tota aplicació Web tenim que utilitzar formularis. Aquests solen estar estructurats mitjançant una graella en la que es mostren els components de formulari (camps d'entrada de dades, de selecció, butons d'opció, etc.) i a l'esquerra una etiqueta associada al camp.

Mitjançant Struts, JSTL i l'ús del tag '<label>' (descuidat per molts desenvolupadors però existent a l'estándar HTML fa molt de temps) la realització d'un component implicaria el següent codi:

```
<tr><td>
   <label for="id" ...>
          <fmt:message key="key.id "/>
   </label>
   <html:text property="id" size="20" maxlength="20"/>
</td></tr>
```

Resulta obvi que a mesura que anem afegint components a la nostra pàgina JSP aquesta es fa feixuga de llegir, ja que incorpora gran quantitat de layout i etiquetes. Per què no un tag que generi tot aquest codi i fós senzill d'utilitzar? Amb el nou tag proporcionat per openFrame el codi equivalent seria:

```
<fwk:text styleId="id" property="id"/>
```

Exemple:

```
<bean parent="textFieldTag">
   <property name="styleId" value="firstname"/>
   ...
   <property name="key" value="forms.accountForm.field.firstname"/>
```

- **accessKey (accés per teclat)**

Per a poder accedir mitjançant el teclat a qualsevol camp, podem usar l'atribut 'accesskey'. Mitjançant HTML podríem crear el següent codi:

```
<label for="nombre"><u>N</u>ombre</label>
<input type="text" accesskey="n" id="nombre" name="nombre">
```

Mode tradicional

openFrame genera de forma automàtica [2] el codi d'accés als components. En conjunció amb el servei de internacionalització defineix quina és la clau que defineix quin és el caràcter a subratllar.

Exemple:

```
<bean parent="textFieldTag">
   <property name="styleId" value="firstname"/>
   <property name="accesskey"
             value="forms.accountForm.field.firstname.accesskey"/>
```

- **mode**

2 Llibreria de generació de access keys: http://www.gerv.net/software/aul/

El mode dels camps permet especificar si un camp "input" ha de ser editable, de lectura, ocult o que no aparegui en el formulari segons el mode de display del formulari. Això permet per exemple que un camp que forma part de la clau primària només sigui editable durant la creació, o mostrar camps calculats només quan es visualitza la informació i no quan s'edita.

El comportament per defecte és que els camps es mostrin editables si el mode del formulari es CREATE o EDIT, i es mostrin de lectura (valor text pla sense component HTML i un hidden amb el valor del camp) si el mode del formulari és INSPECT.

Aquest comportament per defecte pot alterar-se en els camps mitjançant l'atribut 'mode'. El valor que s'ha d'assignar a aquest atribut ha de seguir el patró 'X,Y,Z'; on X és el comportament que tindrà el camp en cas que el mode del formulari sigui 'FormUtils.CREATE_MODE', Y el comportament en el mode edició ('FormUtils.EDIT_MODE') i Z en mode de consulta ('FormUtils.INSPECT_MODE'). Els valors posibles per X,Y i Z són:

- E: editable
- I: inspeccionable (de consulta), amb un camp ocult corresponent
- S: mostrar, igual que 'I' però no es genera camp ocult
- N: no es mostra
- P: present (com 'S' si el valor del camp no és null i com 'N' si el valor del camp és nul)
- H: ocult (no es mostra, però sí es genera un camp ocult)
- R : de lectura (es genera el camp però amb l'atribut 'readonly')
- D: desactivat (es genera el camp amb l'atribut 'disabled')

Una vegada especificat en quin mode funcionarà el formulari, els components apareixeran segons el mode especificat, sense que per això s'hagi de generar codi JSP addicional.

- **required**

Es pot especificar que es mostri una imatge o text al principi de l'etiqueta per indicar que un camp és requerit si especifiquem el valor 'true' per la propietat 'required'.

Aaquesta propietat **no és necessària** especificar-la si en el Servei de Validació s'ha configurat que el camp és obligatori. El tag accedeix al servei i pregunta si el camp associat és requerit. Per a que això funcioni cal que s'especifiqui la propietat 'validationService' al tag.

El comportament proporcionat per openFrame crea el contingut '
'*' abans del tag HTML '<label>' . Aquest comportament està definit al fitxer 'scripts/ajax/behaviour/openFrame-behaviours.js'.

En cas de voler canviar aquest comportament podem accedir al fitxer i fer els canvis necessaris al codi:

Podem configurar l'estil de l'asterisc modificant al fitxer d'estils la següent entrada:

```
label span.required { color: #c32; }
```

- **tooltipKey, tooltipTitleKey y tooltipOptions**

  Aquestes 3 propietats ens permeten especificar l'ajuda contextual als camps.

  - tooltipKey. Clau del literal del missatge a mostrar dins el tooltip
  - tooltipTitleKey. Clau del literal a mostrar si volem mostrar un títol com a capçalera del tooltip
  - tooltipOptions. Opcions aplicades segons la implementació (en el cas actual sota DOM Tooltip).

Per a ser utilitzat el tag 'Configuration' importa el fitxer js 'scrips/ajax/ajaxtags/openFrame-ajaxtags.js'. Si es vol canviar el comportament per defecte es poden fer els canvis necessaris a aquest fitxer.

En aquest codi es genera de forma automàtica el link que mostrarà el tooltip en passar per sobre. Adicionalment podem configurar els estils dels tooltips segons la definició dels estils mostrats a continuació:

```
a.tooltip:link {text-decoration: none; }
a.tooltip:visited {text-decoration: none; }
a.tooltip:hover {text-decoration: none; cursor:help; }
a.tooltip:active {}
```

Un dels estils més útils és 'cursor:help' dins a hover, ja que ens apareixerà el cursor d'ajuda quan passem per sobre de l'etiqueta del component.

NOTA: El cursor d'ajuda no apareix en el gràfic

Configuració del Tag Text

El tag de text permet l'entrada de dades i es basa en el tag de Struts. Per tant, totes les propietats definides a Struts permeten ser definides a openFrame.

A més de les propietats própies als components de formulari i de les de Struts s'han incorporat les següents propietats:

Propietat	Requerit	Valor
autoTab	No	Especificar 'true' si es vol realizar autotabulació en haver introduït el nombre de caràcters especificat a la propietat '**maxLength**' del component.
convertTo	No	Permet definir quines conversions es volen realitzar al valor introduït al camp de forma automàtica en perdre el focus del component. Cal especificar les diferents conversions a aplicar separades per ','. Es proporcionen les següents conversions: • uppercase. Passa tot el contingut a majúscules • lowercase. Passa tot el contingut a minúscules • trim. Elimina els blancs de l'esquerra i dreta del valor. Entre les paraules s'encarrega també de deixar un únic blanc.
showCalendar	No	Indicant 'true' podem especificar que el camp d'edició ha de mostrar un calendari de selecció de data. En cas de que en el servei de validació s'hagi especificat que el camp associat té una validació de data, el tag comprovarà el format definit i serà aquest el que permetrà a openFrame saber quin és el format de la data que ha de copiar en seleccionar una data dins el calendari.

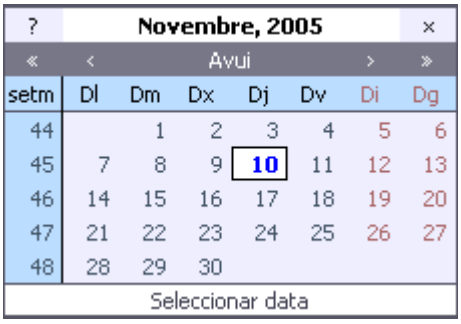
- **showCalendar**

  Per a que el calendari es mostri correctament cal:

  1) Definir el valor 'date' en l'atribut 'depends' dins el fitxer de configuració del 'Servei de Validació'. Definir en 'datePattern' el patró a complir

2) Incorporar el css a utilitzar

```
<style type="text/css">@import url(css/calendars/dynarch/calendar-
blue.css);</style>
```



Podem usar diferents fitxers d'estils o definir un propi. Els fitxers d'estils existents es troben a 'css/calendars/dynarch'.

Els literals del calendari es troben definits en fitxers '.js' que es troben al directori 'scripts/calendars/dynarch/lang/'. El tag 'Configuration' és qui s'encarrega d'afegir la referència al fitxer corresponent a l'idioma de l'usuari.

Integració amb el Servei de Validació

Per a que s'usi un format de data específic, openFrame consulta per la propietat associada al tag (indicada mitjançant l'atribut 'property' del tag) si es tracta d'un objecte de tipus 'java.util.Date'. Si és així, obté el format que s'ha d'aplicar segons l'idioma de l'usuari. Podem configurar quin serà el format de data utilitzat per cada llenguatge definit una nova clau i el seu patró en la propietat 'localeDatePatternsMap' del fitxer 'property-editors.xml' tal i com es mostra a continuació:

```
...
<bean id="customEditors" class="java.util.HashMap">
<constructor-arg>
   <map>
        <entry key="java.util.Date">
                <bean
class="net.opentrends.openframe.services.i18n.spring.beans.propertyeditors.Cust
omDateEditor">
                      <property name="i18nService" ref="i18nService"/>

                             <property name="localeDatePatternsMap">

<map>
<entry>
```

```
<key><value>es</value></key>
<value>dd/MM/yyyy</value>
                              </entry>
                              <entry>
                         <key><value>en</value></key>
                  <value>MM/dd/yyyy</value>
        </entry>

                  </map>
    </property>
...
 </bean>
```

Configuració del Tag 'Password'

El tag de password permet definir les mateixes propietats que el tag 'Text'.

Exemple:

```
<bean parent="passwordFieldTag">
       <property name="mode" value="E,I,I"/>
       <property name="styleId" value="password"/>
       <property name="key"          value="forms.accountForm.field.password"/>

</bean>
```

Configuració del Tag 'TextArea'

Propietat	Requerit	Valor
rows	Sí	Número de files
cols	Sí	Número de columnes
decoratorProperties	No	Permet especificar la generació d'una àrea amb justificats, formats, ...

- **decoratorProperties**

  Permet especificar la generació d'un textarea en el que es permetin justificats, formats, colors, etc. El contingut aquí introduit serà enviat com HTML.

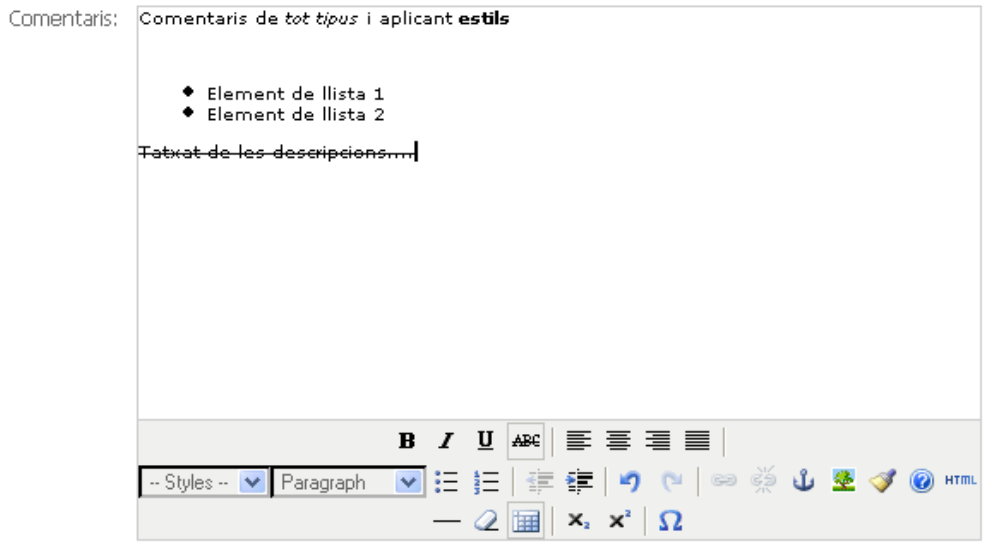
  Per especificar els decoratorProperties usarem els següents valors:

  - theme: 'advanced' o 'simple' (si volem més o menys opcions)
  - mode: 'exact'

Exemple:

```
<bean parent="textAreaFieldTag">
   <property name="styleId" value="comments"/>                    <property
name="key" value="forms.accountForm.field.comments"/>          <property
name="rows" value="10"/>                                       <property
name="cols" value="40"/>                                       <property
name="decoratorProperties">
          <map>
                <entry>
                  <key><value>theme</value></key>
                  <value>advanced</value>
                </entry>
                <entry>
                  <key><value>mode</value></key>
                  <value>exact</value>
                </entry>
</map>
   </property>
</bean>
```

Mitjançant l'exemple a dalt mostrat tindríem un component com el mostrat a continuació:



Configuració del Tag 'Select'

Permet definir les mateixes propietats que les definides per Struts i les propietats generals a tots els tags d'entrada de formulari definides anteriorment.

Adicionalment es proporcionen 2 funcionalitats prou importants:

a) Generació d'opcions segons una query de base de dades
b) Generació de les opcions segons el valor escollit en un altre select de forma dinàmica sense refresc de la pàgina

- **Generació de les opcions segons query de la base de dades**

  Si volem mostrar una llista d'opcions que prové d'una base de dades openFrame openFrame ofereix la possibilitat de definir la font de les dades.

  Per això cal que seguim els següents pasos:

  1) Definir l'adaptador que s'usarà basat en Hibernate

```
<bean name="defaultOptionBaseHibernateAdapter"
       class="net.mlw.vlh.adapter.hibernate3.Hibernate30Adapter">
       <property name="sessionFactory" ref="sessionFactory"/>
       <property name="defaultNumberPerPage" value="5"/>
       <property name="defaultSortColumn" value="id"/>
       <property name="defaultSortDirection" value="asc"/>
       <property name="removeEmptyStrings" value="true"/>
</bean>
```

  El seu ús és equivalent al definit pel Servei de Llistats.

  2) Gestors de les llistes

```
<bean name="defaultOptionValueListHandler"
       class="net.mlw.vlh.DefaultValueListHandlerImpl">
       <property name="config.adapters">
              <map>
                     <entry key="categoriesList">
                            <bean parent="defaultOptionBaseHibernateAdapter">
                                   <property name="hql">
                                          <value>
                                                 FROM
net.opentrends.openframe.samples.jpetstore.model.Category
                                                 AS vo WHERE 1=1
                                                 /~descn: AND vo.descn LIKE
{descn} ~/
                                                 /~sortColumn: ORDER BY
vo.[sortColumn] [sortDirection]~/
                                          </value>
                                   </property>
                            </bean>
```

```
                </map>
        </property>
</bean>
```

En la propietat 'config.adapters' definirem les diferents queries a executar per a recuperar les llistes. Aquestes queries es troben definides en HQL, pel que obtenim objectes.

- key. Nom de la llista (aquest nom serà utilitzat des de la definició del tag)
- hql. Definició de la consulta Hibernate per a obtenir la llista de valors

```
<entry key="nomLlista">
<bean parent="defaultOptionBaseHibernateAdapter">
        <property name="hql">
        ...
```

3) Definir les següents entrades:

```
<bean id="defaultOptionValueListHandlerHelper"
class="net.mlw.vlh.web.mvc.ValueListHandlerHelper">
<property name="valueListHandler">
        <ref bean="defaultOptionValueListHandler" />
</property>
</bean>


<bean name="defaultOptionListSource"
class="net.opentrends.openframe.services.web.taglib.util.options.vlh.hibernate3
.VlhOptionListSourceImpl">
<property name="valueListHandlerHelper">
        <ref bean="defaultOptionValueListHandlerHelper" />
</property>
</bean>
```

4) Definir la informació de la font de la nostra llista

- optionListName: Referència a la definició de la llista definida al pas 2
- optionLabelName: Atribut de l'objecte recuperat que servirà d'etiqueta de l'opció
- optionLabelProperty: Atribut de l'objecte recuperat que servirà de valor de l'opció

```
<bean id="categoriesOptionListSource" parent="defaultOptionListSource">
              <property name="optionListName" value="categoriesList"/>
              <property name="optionLabelName" value="name"/>
              <property name="optionLabelProperty" value="id"/>

</bean>
```

5) Incorporar la llista al servei d'opcions

```
<bean id="optionsListService"
class="net.opentrends.openframe.services.web.taglib.util.options.OptionsListSer
viceBase">
<property name="optionsListSources">
       <map>
              <entry key="categoriesList">
              <ref bean="categoriesOptionListSource"/></entry>

              <entry key="animalsList">
              <ref bean="animalsOptionListSource"/></entry>

       </map>
</property>
</bean>
```

6) Per últim definir a la propietat 'optionsListService' del tag una referència al servei d'opcions (propietat 'optionsListService') i afegir el nom de la llista a la propietat 'optionsListName':

```
<bean id="selectFieldTag"
class="net.opentrends.openframe.services.web.struts.taglib.forms.fields.SelectF
ieldTag">
...
       <property name="optionsListService" ref="optionsListService"/>
...
```

```
<bean parent="selectFieldTag">
       <property name="key" value="forms.accountForm.field.preferredCategory"/>
                              <property name="styleId"
value="preferredCategoryId"/>              <property name="optionsListName"
value="categoriesList"/>
</bean>
```

- **Generació de les opcions segons el valor escollit en un altre select**

En cas de voler presentar la llista d'opcions d'un select segons la selecció realitzada a un altra selecció, definirem a més del definit en la secció anterior un nou atribut 'selectFieldSource' en el que definirem:

- source. Nom de la propietat 'styleId' utilitzada en el select font
- paramName. Nom del paràmetre de la query definit entre {}.

Per exemple, imaginem que volem mostrar la llista d'animals segons la categoria escollida.

Podem definir la hql de la llista 'animalsList' tal i com es mostra a continuació:

```
<entry key="animalsList">
<bean parent="defaultOptionBaseHibernateAdapter">
        <property name="hql">
                <value>
                FROM
        net.opentrends.openframe.samples.jpetstore.model.Product
                AS vo WHERE 1=1
                /~category: AND vo.category.id LIKE {category} ~/
                /~sortColumn: ORDER   BY vo.[sortColumn]
[sortDirection]~/
                </value>
        </property>
</bean>
</entry>
```

En aquesta hql apareix el paràmetre {category} dins el LIKE que és el que ens permetrà lligar que se'ns retornin els productes amb la categoria indicada.

Per a que s'envii com a criteri valor dins {category} el valor escollit a la select source definirem com a 'paramName' el valor 'category'.

```
<bean parent="selectFieldTag">
        <property name="key" value="forms.accountForm.field.preferredAnimal"/>
                                <property name="styleId"
value="preferredAnimalId"/>                     <property name="optionsListName"
value="animalsList"/>                 <property name="selectFieldSource">
        <map>
              <entry>
                <key><value>source</value></key>
                <value>preferredCategoryId</value>
              </entry>

              <entry>
                <key><value>paramName</value></key>
```

```
                <value>category</value>
              </entry>
    </map>

          </property>
    </bean>
```

Configuració del Tag 'Checkbox'

Permet definir les mateixes propietats que les definides per Struts i les propietats generals a tots els tags d'entrada de formulari definides anteriorment.

Exemple:


```
<bean id="checkboxFieldTag"
class="net.opentrends.openframe.services.web.struts.taglib.forms.fields.CheckboxFi
eldTag">
          <property name="i18nService" ref="i18nService"/>
          <property name="validationService" ref="webValidationService"/>
          <property name="layout" value="true"/>
</bean>
```

```
<bean parent="checkboxFieldTag">
   <property name="styleId" value="lower18"/>
   <property name="key" value="forms.accountForm.field.lower18"/>
</bean>
```

Configuració del Tag 'File'

Aquest tag permet especificar l'ús d'un fitxer de upload

Veure 'Servei de Upload de Fitxers' per a més referència de la configuració necessària.

El tag requereix que es configuri la propietat 'encType' del tag 'Form' amb el següent valor:

"multipart/form-data"

Exemple:

```
<bean parent="fileFieldTag">
          <property name="styleId" value="file"/>
          <property name="mode" value="E,I,I"/>
          <property name="key" value="forms.fileForm.field.file"/>
</bean>
```

2.2.5. Configuració dels tags de Llistats

Veure document 'Servei de Llistats'.

2.3. Utilització del Servei

2.3.1. Ús dels tags a les pàgines JSP

L'ús de cadascun dels tags implica 2 pasos:

1) Definició de la referència a la llibreria de tags

Des de la versió JSP 1.2 no és necessari configurar al fitxer web.xml la llibreria i podem fer referències directes mitjançant url al jar, tal i com es mostra a continuació:

```
<%@ taglib prefix="fwk"  uri="http://www.opentrends.net/openframe/open-layout" %>
```

2) Definició del tag a la pàgina JSP

Una vegada definida la referència a la llibreria i el seu prefix 'fwk' podem incorporar qualsevol tag mitjançant '<fwk:nomTag>'.

Per tots els tags d'entrada de dades de formulari es definiran els atributs:

Propietat	Requerit	Valor
styleId	Sí	Identificador del camp. Aquest identificador és el que correspon a la propietat 'styleId' en el fitxer de configuració del tag
property	Sí	Nom del atribut del bean associat al formulari que volem mostrar. (l'associació del bean es fa mitjançant la propietat 'pojoClass' del fitxer de configuració de l'acció)

Exemple:

```
<fwk:text styleId="firstname" property="firstname"/>
```

En la següent taula es dona un resum dels tags permesos i la correspondència amb la seva classe que s'ha de definir en el fitxer de configuració:

Tag	Classe

Tag	Classe
fwk:actionImage	net.opentrends.openframe.services.web.struts.taglib.forms.fields.ActionImageTag
fwk:checkbox	net.opentrends.openframe.services.web.struts.taglib.forms.fields.CheckboxFieldTag
fwk:configuration	net.opentrends.openframe.services.web.struts.taglib.configuration.ConfigurationTag
fwk:file	net.opentrends.openframe.services.web.struts.taglib.forms.fields.FileFieldTag
fwk:form	net.opentrends.openframe.services.web.struts.taglib.forms.fields.FormTag
fwk:options	net.opentrends.openframe.services.web.struts.taglib.forms.fields.OptionsFieldTag
fwk:password	net.opentrends.openframe.services.web.struts.taglib.forms.fields.PasswordFieldTag
fwk:radio	net.opentrends.openframe.services.web.struts.taglib.forms.fields.RadioFieldTag
fwk:select	net.opentrends.openframe.services.web.struts.taglib.forms.fields.SelectFieldTag
fwk:submit	net.opentrends.openframe.services.web.struts.taglib.forms.fields.SubmitTag
fwk:text	net.opentrends.openframe.services.web.struts.taglib.forms.fields.TextFieldTag
fwk:textarea	net.opentrends.openframe.services.web.struts.taglib.forms.fields.TextAreaFieldTag

2.3.2. Tag de pestanyes

El tag de pestanyes té un comportament especial ja que no necessita cap element de configuració dintre dels fitxers d'Spring, sino que s'inclou directament al JSP. La sincronització del contingut de cada una de les pestanyes es produeix amb AJAX (ajax-tags).

Exemple:

```
<fwk:tabPanel
    panelStyleId="navcontainer"
    contentStyleId="tabContent"
    currentStyleId="active"
    postFunction="initializeMenus">

  <fwk:tab captionKey="tabs.petstore"
    baseUrl="${pageContext.request.contextPath}/AppJava/ptop/pagetab1.do"
defaultTab="true"/>
  <fwk:tab captionKey="tabs.exit"
baseUrl="${pageContext.request.contextPath}/AppJava/ptop/pagetab2.do"/>
  <fwk:tab captionKey="tabs.admin"
baseUrl="${pageContext.request.contextPath}/AppJava/ptop/pagetab3.do"/>

</fwk:tabPanel>
```

L'element base és el "tabPanel", que representa l'agrupació de pestanyes. Per cada pestanya apareix un "tab" amb la pàgina a carregar, que en temps d'execució es convertirà en un element "li" d'HTML. El look&feel de les pestanyes s'assigna per CSS.

Exemple:

```
<style type="text/css">
<!--

#navcontainer
{
margin-top: 2px;
padding: 0px;
height: 20px;
}

#navcontainer ul li
{
display: block;
float: left;
text-align: center;
padding: 0px;
margin: 0px;
}
#navcontainer ul li a
{
width: 44px;
height: 18px;
border-top: 1px solid #000000;
border-left: 1px solid #000000;
border-right: 1px solid #000000;
padding: 0px;
margin: 0px 0px 0px 0px;

color: #000000;
background: #CCCCCC;
text-decoration: none;
display: block;
text-align: center;
font-family: Arial, Verdana,  Helvetica, sans-serif;
font-size: 10px;
}

#navcontainer ul li a:hover
{
color: #000000;
background: #CCCCCC;
}

#navcontainer ul li a:active
{
background: #F3D299;
border-top: 1px solid #000000;
border-left: 1px solid #000000;
border-right: 1px solid #000000;
color: #000000;
font-weight: bold
}

#navcontainer ul li#active a
{
background: #F3D299;
border-top: 1px solid #000000;
border-left: 1px solid #000000;
border-right: 1px solid #000000;
color: #000000;
```

```
}

-->
</style>
```

La següent taula mostra una llista de cadascún dels atributs que es ponen als nodes que formen el tag de pestanyes:

(1) fwk:tabPanel

Paràmetre	Descripció	Requerit
panelStyleId	El ID del panell ( ) del tab panel	Si
contentStyleId	El ID del panell ( ) del contingut del tab panel	Si
currentStyleId	El ID del CSS a fer servir per la pestanya activa [per defecte='current']	Si
postFunction	La funció javascript a executar després que la pestanya s'ha carregat	No
emptyFunction	La funció javascript a executar si hi ha una resposta del servidor buida	No
errorFunction	La funció javascript a executar si hi ha un error de servidor	No

(2) fwk:tab

Paràmetre	Descripció	Requerit
baseUrl	La URL de la pàgina de contingut que es demanarà	Si
captionKey	La clau de l'etiqueta a mostrar (i18n)	Si
defaultTab	Si és la pestanya per defecte [true|false]	No
parameters	Paràmetres separats per comes que es pasaran a la crida de la URL.	No

2.3.3. Especificació del Mode del Formulari

En la configuració dels components d'entrada de formulari s'ha explicat cóm podem mostrar-los de forma diferents segons el mode del formulari. Aquest mode de formulari pot canviar-se mitjançant la crida:

FormUtils.setFormDisplayMode(request, actionForm,mode);

On mode pot ser un dels següents valors:

- FormUtils.INSPECT_MODE. Mode de consulta
- FormUtils.EDIT_MODE. Mode d'edició
- FormUtils.CREATE_MODE. Mode de creació

NOTA:
openFrame, dins la seva classe 'ActionExtendedSupport' (classe pare de les nostres action) efectua aquesta tasca per a que no ens haguem de preocupar. Per a això utilitza un resolver que segons el reqCode que arribi per parámetre decidirà el model del formulari. A continuació es mostra una taula amb els modes usats segons el reqCode:

```
public static final String EDIT_METHOD_PREFIX = "edit";
```

```
public static final String SEARCH_METHOD_PREFIX = "search";
public static final String NEW_METHOD_PREFIX = "create";
public static final String DELETE_METHOD_PREFIX = "delete";
public static final String SAVE_METHOD_PREFIX = "save";
public static final String VIEW_METHOD_PREFIX = "view";
```

reqCode	Mode utilitzat
edit	FormUtils.EDIT_MODE
search	FormUtils.INSPECT_MODE
create	FormUtils.CREATE_MODE
view	FormUtils.INSPECT_MODE
En altres casos	FormUtils.EDIT_MODE

2.3.4. Estructuració del Contingut (Layout)

Els tags de formulari i inputs d'entrada, cóm s'ha comentat prèviament, generen de forma automàtica el layout (table, td, tr) per incrementar la productivitat del desenvolupament de les pàgines i millorar la seva mantenibilitat (no apareixen dins la pàgina JSP pel que aquesta resulta molt més simple). Aquest comportament es pot deshabilitar mitjançant l'ús de la propietat 'layout' de cadascun dels components, però no es recomana el seu ús.

Per defecte, el comportament és ubicar els components d'esquerra a dreta segons estiguin ubicats a la pàgina, però en molts casos convé posicionar-los de forma específica amb columnes, cel·les, etc.

Per aquest objectiu, openFrame proporciona varis tags JSP que ens poden ajudar a estructurar les nostres pàgines:

- fwk:grid. Permet especificar que tots els tags inclosos siguin renderitzats en una taula amb el nombre indicat de columnes

  Les propietats del tag són:

Propietat	Requerit	Valor
borderSpacing	No	Espai de les vores. Per defecte és 0.
cols	No	Nombre de columns. Per defecte 2.
space	No	Indicar true per si volem incloure una cel·la buida entre cada cel·la
styleClass	No	Estil del grid
width	No	Amplada del grid

- fwk:row. Inserir una fila

Les propietats del tag són:

Propietat	Requerit	Valor
space	No	Indicar true per si volem incloure una cel·la buida entre cada cel·la

- fwk:cell

  Les propietats del tag són:

Propietat	Requerit	Valor
colspan	No	Colspan de la cel·la
height	No	Alçada de la cel·la
styleClass	No	Estil CSS de la cel·la
width	No	Ample de la cel·la

- fwk:column. Inserir una columna. Tots els tags inclosos es mostraran de forma vertical

Propietat	Requerit	Valor
styleClass	No	Estil CSS de la columna

- fwk:line. Té el mateix comportament que el tag 'row' amb la diferència que els tags continguts en diferents línies s'alinien (com una taula dintre d'una cel.la)

Propietat	Requerit	Valor
space	No	Indicar true per si volem incloure una cel·la buida entre cada cel·la

2.3.5. Comentaris Específics a alguns Tags

En l'ús del tag 'select' es recomana l'ús de JSTL per a la generació de les opcions incloses.

Exemple:

```
<fkw:select styleId="..." property="...">
    <c:forEach items="${nombreLista}" var="item">
      <fwk:option value="${item}">${item.descripcion}</fwk:option>
    </c:forEach>
  </select>
```

2.4. Eines de Suport

2.4.1. Debug de les Pàgines

En l'ús de les nostres pàgines és convenient sempre conéixer si estan ben construïdes. Tal i com es comenta al document 'Entorn de Desenvolupament' mitjançant l'extensió 'Web Developer' i 'Javascript Debugger' podem esbrinar i detectar problemàtiques de les nostres pàgines.

2.5. Integració amb Altres Serveis

2.6. Preguntes Freqüents

3. Exemples

4. Annexos

4.1. JSTL

Des de fa un temps existeix **JSTL** (JSP Standard Tag Library). Aquesta llibreria de tags defineix algunes de les tasques bàsiques que necessitem a les nostres pàgines JSP (condicionals, presentació de valors, internacionalització, ...). Si bé no ofereix totes les necessitats que requerim d'una aplicació Web sí ofereix la base del que prèviament es feia amb codi Java o els tags de Struts.

Es recomana l'ús dels tags JSTL de 'core' i 'fmt' que es poden referenciar dins les nostres pàgines com:

```
<%@ taglib prefix="c"   uri="http://java.sun.com/jstl/core" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jstl/fmt" %>
```

Podeu veure un resum de JSTL a la url 'http://www.1x4x9.info/files/jstl/html/online-chunked'.

4.2. Altres Tags

Tot i l'oferta de tags de openFrame, existeixen altres necessitats, com per exemple l'ús de pestanyes.

En aquest apartat s'ofereixen una llista de components disponibles a la comunitat que poden resoldre aquestes necessitats:

- Tags de Crumbs

Per a mostrar el típic tag d'enllaços a on ens trobem dins la pàgina podem usar el tag de Crumbs de 'Struts Layout'.

[Categorías > **Editar categoría**]

Consultar 'http://struts.application-servers.com/doc/tags/crumbs.html' per conéixer la seva utilització.

- Tags de Pestanyes

Per a utilitzar pestanyes es recomana la utilització del tag de 'Ditchnet'[3]

4.3. Aspectes d'Usabilitat

Aquesta secció pretén donar una visió general del quins aspectes s'han tingut en consideració en els components openFrame per a complir amb normes bàsiques d'usabilitat i accessibilitat.

Algunes de les guies proporcionades poden resultar óbvies però considerem la necessitat de fer-les esmentar per a tenir en consideració. En qualsevol cas, moltes de les guies explicades es troben implementades per openFrame en els seus tags, pel que no s'hauran de tenir en consideració per al desenvolupament de les aplicacions Web. Si més no, també és el nostre propósit mostrar per què s'han pres determinades decisions basades en les guies aquí establertes.

A continuació es detallen les guies utilitzades:

1) La descripció del component d'entrada hauria de precedir al component (excepte els radio i els checkbox)
2) Usar l'etiqueta 'label' i l'atribut 'for' per a fer referència al component que descriu.
   (1) i (2) són realitzats pels tags openFrame.

3) Els camps de tipus text haurien de ser seleccionats quan reben el focus

   openFrame permet especificar la propietat 'selectOnFocus' per a lligar aquest comportament als nostres components. Per defecte es troba a 'true'.

4) Els camps haurien de definir el seu tamany màxim mitjançant les propietats 'maxLength', 'width', 'size' i 'cols'

   Aquestes propietats han de ser definides pels desenvolupadors.

5) En els camps requerits haurem d'utilitzar alguna representació per indicar que són obligatoris. Si es fa servir una icona o símbol (*) indicar a la pàgina què significa

openFrame mostra, per defecte, un asterisc a l'esquerra de l'etiqueta del component si el seu atribut associat ha estat definit com a obligatori dins el Servei de Validació.

6) Quan anem a entrar dades a un component hauríem de mostrar alguna decoració que indiqui que estem en aquell component

   Per exemple podem usar el següent estil (basat en 'theserverside.com'):

```
form input, form textarea, form select {
```

3 http://209.61.157.8:8080/taglibs/

```
    color: #666;
}
```

O bé:

```
input.input-text, textarea { border: 1px solid #859085; background: transparent;
padding: 1px 3px; }
input.hover, textarea.hover { border-color: #000; background-color: #ffe; }
input.active, textarea.active { border-color: #f00; background-color: #fff; }
input.input-button, button, select { border-width: 1px; padding: 0px; }
input.input-text, input.input-button, button, select, textarea { font: 90%
tahoma,verdana,sans-serif; }
input.input-disabled { background-color: #ccc; border-color: #999; color: #777; }
```

4.4. CSS (Cascade Style Sheets)

Tot i que no és propósit de openFrame l'oferir un tutorial en detall de qué són els CSS i cóm fer-los servir es considera bàsic el seu coneixement.

A continuació s'ofereix un resum de quins aspectes es consideren important a tenir en compte:

4.4.1. Importació de fitxers d'estils

Podem agrupar l'ús de varis fitxers d'estils mitjançant la sentència '@import'.

```
<style type="text/css" media="all">
@import "file1.css";
@import "file2.css";
</style>
```

4.4.2. Utilització de ids i classes

Mitjançant l'ús de identificadors de classe (incorporant '.' al davant del nom de la classe) podem definir quin serà l'estil a aplicar a tots els tags que incorporin aquest nom de class (en els tags openFrame correspon a la propietat 'styleClass').

```
.required {font-family: Verdana; font-size: 12pt; color: red; }
```

Aquest estil s'aplicarà a tots els components que tinguin com a class='required', siguin divs, spans, ...

En cas que volem aplicar un estil a un únic component farem servir identificadors. És a dir, partint del id assignat al component (en openFrame l'atribut 'styleId' dels components), podem assignar un estil usant '#':

```
#header {width: 90%; background: white; font-size: 20px; color: purple; }
```

```
<div id="header">...</div>
```

Sempre podem definir els estils al mateix component (atribut 'style') o sobreescriure els definits al fitxer d'estil:

```
<span class="caution" style="color: green">text</span>
```

Si volem definir el mateix estil a diferents ids o classes podem separar cadascun d'ells amb espai tal i com es mostra en el següent exemple:

```
p b {color: red; }
```

Inclús podem definir que tots els components inclosos a un id o class determinat actuin de forma determinada:

```
div#navigation a {color: white; }
```

En aquest cas, s'aplicarà l'estil a tots els '<a href...' que estiguin inclosos a un div amb id='navigation'

4.4.3. Efectes Comuns

En la següent taul es mostra un resum d'algunes de les propietats més útils a definir:

Efecte	Configuració
Text en itàlica	font-style: italic; (tornar a normal amb font-style:normal)
Text en negreta	font-weight: bold; (tornar a normal amb font-weight:normal)
Text en normal	
Capitalització	• Tot a majúscules: text-transform: uppercase; • Tot a minúscules: text-transform: lowercase;

	• Primera lletra a majúscules: text-transform: capitalize; • Normal:text-transform: normal;
Enllaç sense decoració	**a** {text-decoration: none; }
Marges	margin: 20px; margin-left: 2px; margin-top: 80px; margin-right: 45px; margin-bottom: -5px; **IMPORTANT**: Des de que va sortir HTML el body incorpora un espai addicional que podem eliminar amb: **body** {margin: 0px; padding: 0px; }
Padding	S'aplica entre els elements padding: 6px; padding-bottom: 2px; padding-left: 18px;
Color de Background als elements	background-color: green; background-color: rgb(0,108,64);
Imatge de background als elements	background-image: url(images/required.gif);

4.4.4. Pseudo Classes

Tot i que podem definir selectors o estils a la majoria d'elements, hi ha parts que no són elements en sí sinó parts d'aquests. Mitjançant pseudo classes podem accedir a aquests elements.

Com a guia de openFrame es proporciona al fitxer d'estils una pseudo classe per presentar els components d'entrada de diferent forma quan ubiquem el focus.

```
form input, form textarea, form select {
    padding-left: 4px;
    color: #666;
}
```

```
/* Warning: IE doesn't support pseudo-class :focus */
form input:focus, form textarea:focus, form select:focus {
    border-bottom: #ffdead solid 2px;
    border-right: #ffdead solid 2px;
    border-left: #c07300 solid 2px;
    border-top:  #c07300 solid 2px;
    color: #000;
}
```

NOTA: Internet Explorer no suporta pseudo classes