Webalkalmazás fejlesztés Java EE környezetben NetBeans segítségével:

JavaServer Faces 1.2
AJAX

Varga Péter
peter.varga@sun.hu
Zsemlye Tamás
tamas.zsemlye@sun.com

Áttekintés

- Hagyományos webalkalmazás-fejlesztés
- JavaServer Faces
- A JSF és az AJAX kapcsolata
- A JSF barátsága a vizuális fejlesztőeszközökkel
- A NetBeans JSF támogatása

Hagyományos webalkalmazás fejlesztés JAVA segítségével

Servlet specifikáció

- Többszálú kérésfeldolgozási modell
- Skálázhatóság
- J2EE szabvány része, sok-sok elérhető szolgáltatás
- Objektum orientált szemlélet
- Hátránya, hogy JAVA tudás szükséges

Hagyományos webalkalmazás fejlesztés JAVA segítségével

JavaServer Pages specifikáció

- HTML közeli leírónyelv, mégis dinamikus tartalom
- A JSP szervletté alakul futási időben
- Scriptlet (nem szerencsés)
- Action
- Tag libraries (JSP 2.0 óta beépített: JSTL)
- JavaBeans szabvány
- Expression Language (JSP 2.0)

JavaBeans™ komponensek

Ez egy Java osztály, mely:

- Teljesen egységbe zárt (nincs publikus változója)
- Rendelkezik publikus, paraméterek nélküli konstruktorral
- Tulajdonságai (bean-property) get, és set metódusok által definiáltak.

JavaBeans™ komponensek

```
public class Customer {
    private String myName;
    private Address myAddress;
    public String getName() {
        return myName;
    }
    public void setName(String name) {
        myname = name;
    }
    public Address getAddress() {
        return myAddress;
    }
```

JavaBeans™ komponensek

```
public class Address {
    private String city;
    private String street;
    public String getCity() {
        return city;
    }
    public String getStreet() {
        return street;
    }
}
```

Expression Language

```
session.getAttribute("cust").getAddress().ge
  tCity();
```

```
${sessionScope.cust.address.city}
```

```
${1 + (3 * 4)}
```

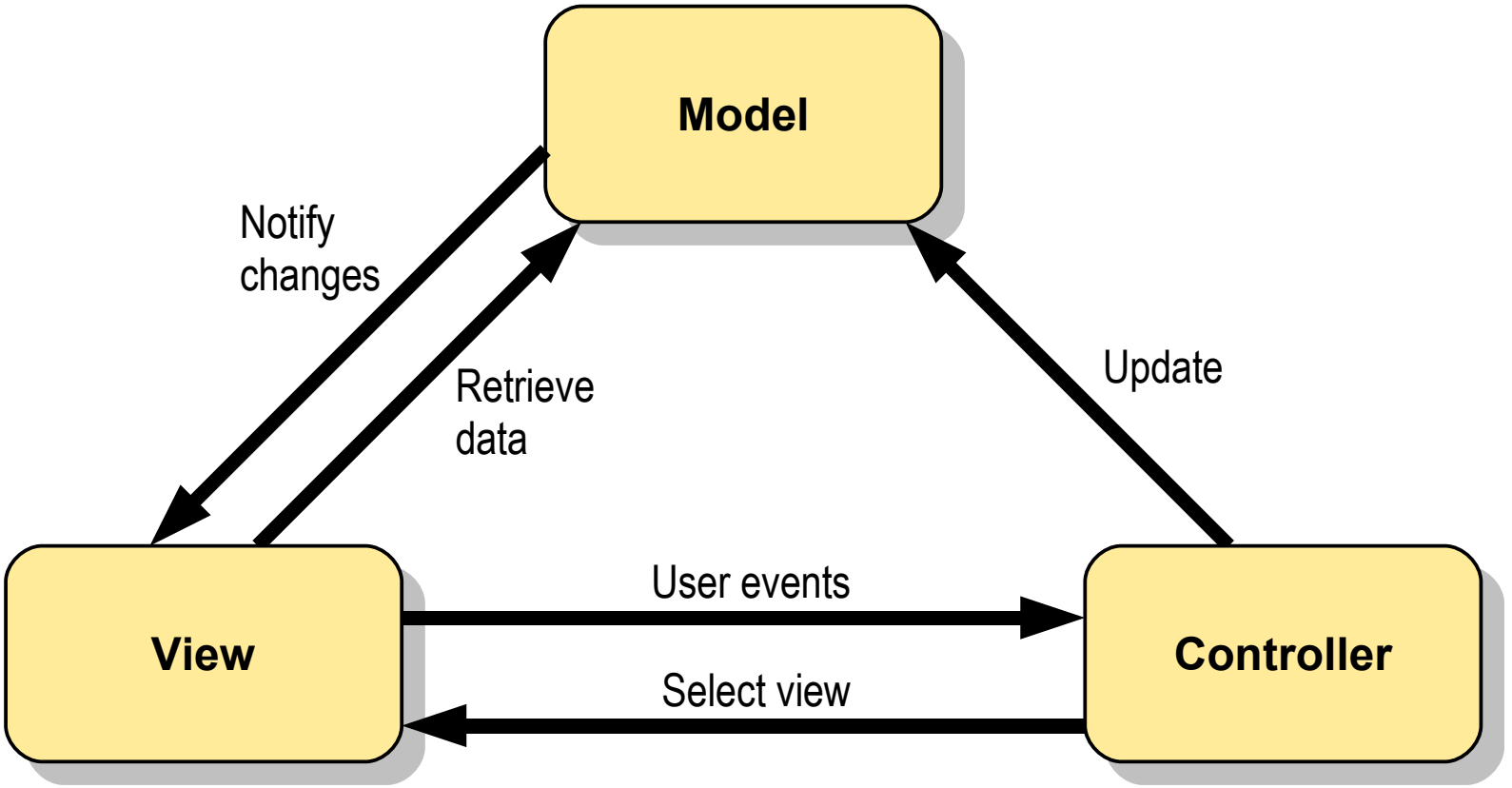
```
${not empty myBean.customerList}
```

```
${applicationScope.myMap['list'][param.i
  ]}
```

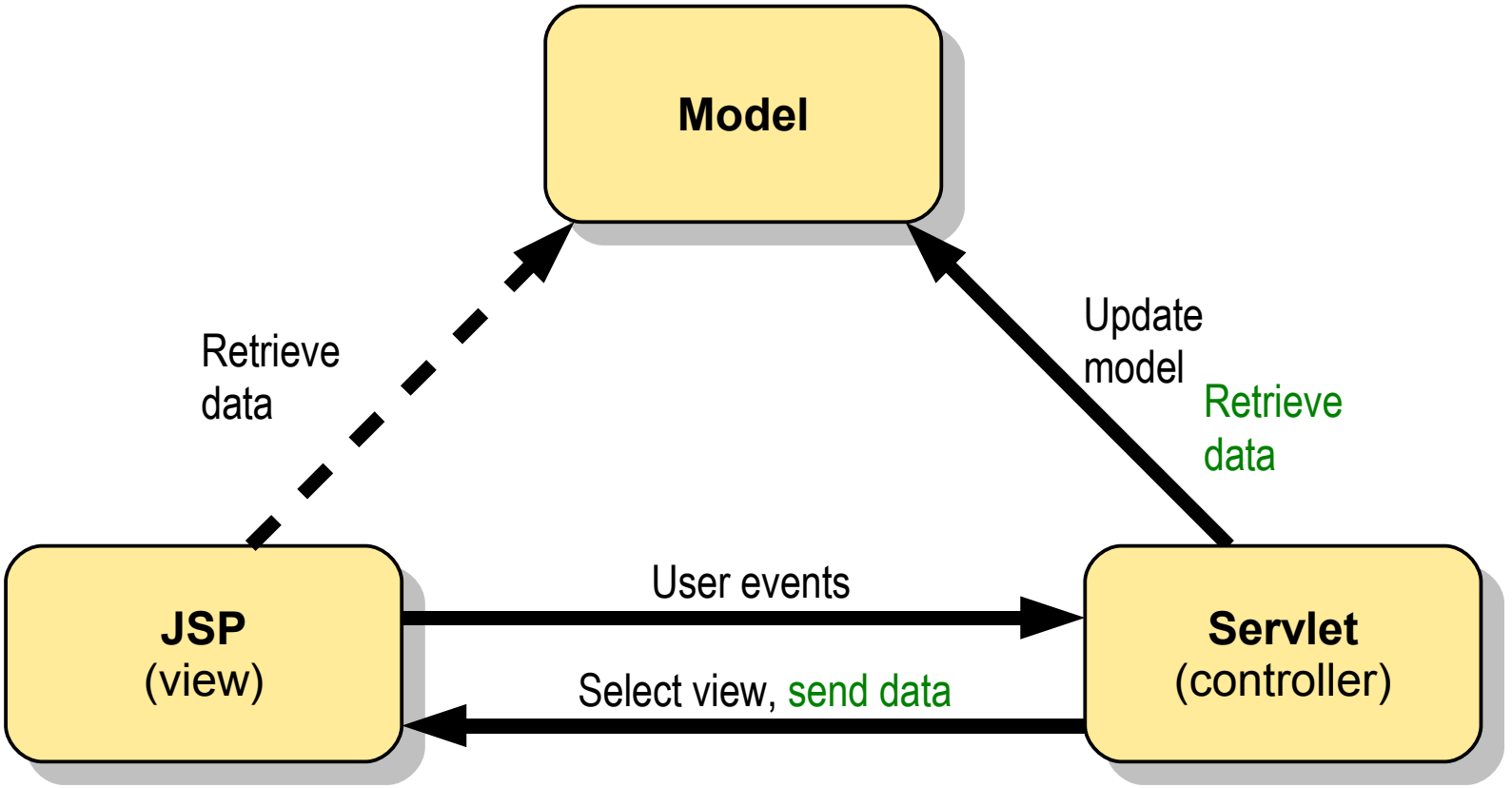
Java™ alapú webalkalmazás fejlesztés szakaszai

- Model 1 architektúra - káosz
- Model-View-Controller web adaptációja - Model 2
- Web MVC keretrendszerek
- Szabványos keretrendszer – JavaServer Faces (J2EE 5.0 óta)

Model-View-Controller



Model 2 architektúra



Model 2 keretrendszerek

- Tervezési minták alkalmazása – kézbentartható webfejlesztés
  - Front Controller
  - Application Controller
  - Command
  - Context Object
- Kész keretrendszerek használata
  - Sun's J2EE blueprints Web Application Framework
  - Apache Struts (de facto szabvánnyá vált)
  - Spring WebMVC
  - Sok egyéb...
  - JavaServer Faces

JavaServer Faces

Alapelvek

- Az alapkoncepció – az absztrakció ereje
  - > Komponens modell – feketedoboz jellegű építőelemek
  - > Komponensek adatmodellel való összekötése - EL
  - > Navigáció – egy irányított gráf (ha akarjuk)
- Futási idejű teljesítmény
  - > Több implementáció – válasszuk a leggyorsabbat
  - > Minden ponton bővíthető/lecserélhető felépítés: lecserélhető az implementáció tetszőleges része egy saját, optimalizált verzióval

JavaServer Faces (folytatás)

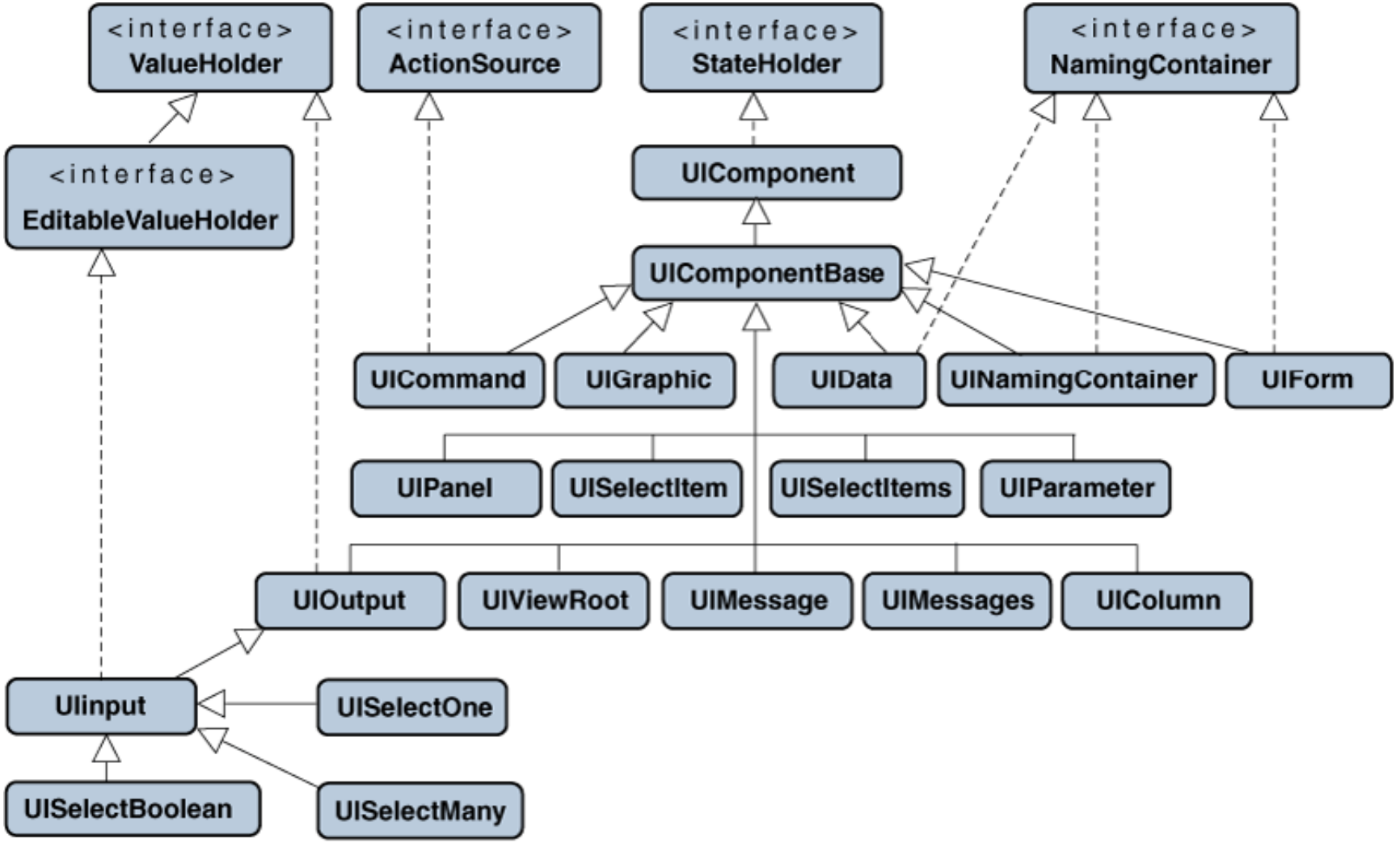
Alapelvek

- Elkülöníthető szerepkörök
  - > Oldal készítő
  - > Komponens fejlesztő
  - > Alkalmazás fejlesztő
  - > Fejlesztőeszköz készítő

JSF komponensek

- Egy komponens egy önálló felhasználói interakciót reprezentáló egység, például:
  - Logikai érték beállítás
  - Szövegbeírás
  - Egy lista elemének kiválasztása
  - Egy lista több elemének kiválasztása
- Függetlenek az adatátviteli protokolltól
- Függetlenek a megjelenéstől
- A komponensek élete egy kérés feldolgozásáig tart
- Saját állapotuk van – ez elmenthető és visszaállítható

JSF komponens hierarchia

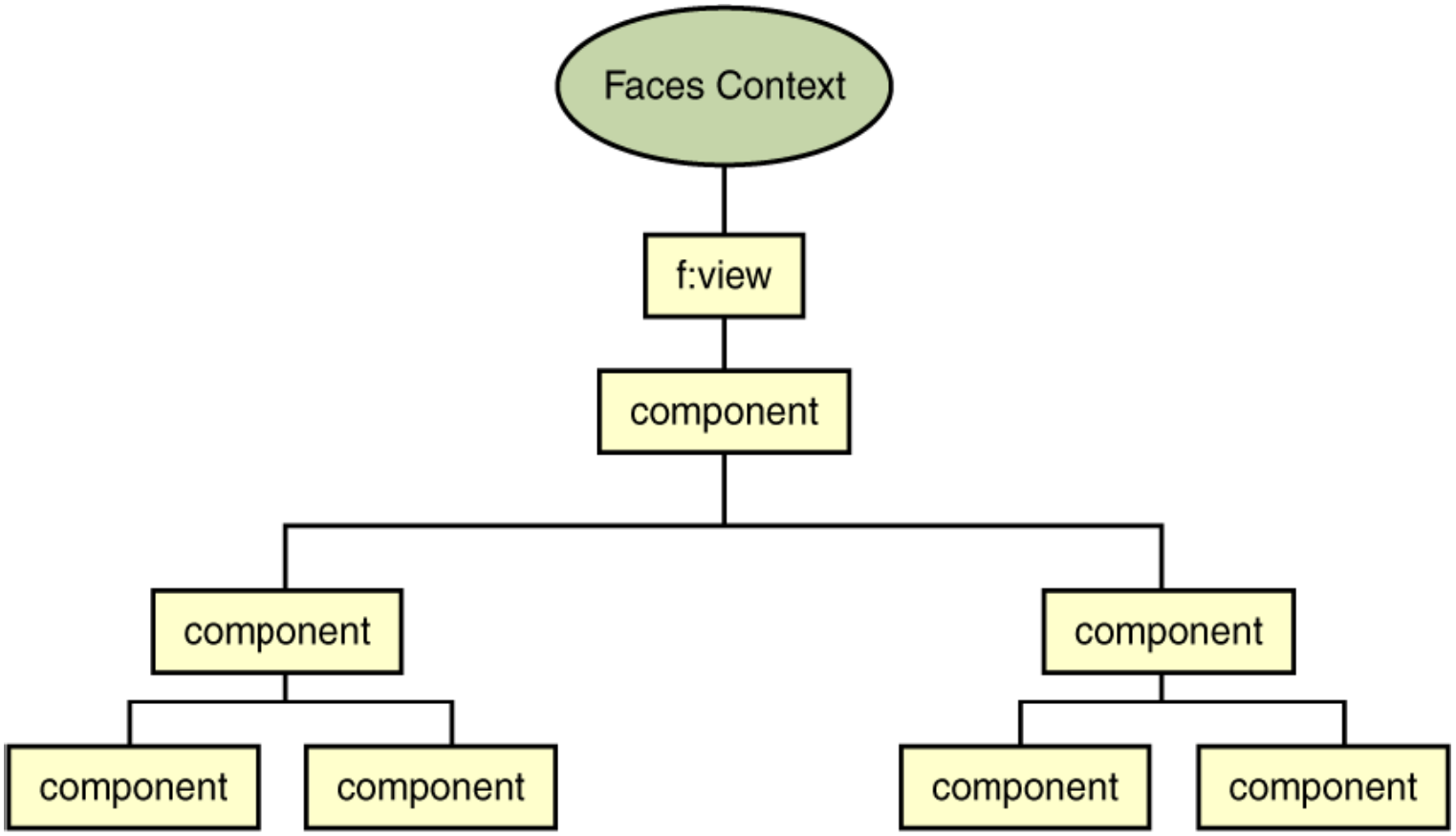


JSF komponensek

- A felhasználó által ismert komponensek az absztrakt komponensek és a rendererek együtteséből állnak össze, például:
  - UIData + Table = DataTable
  - UIInput + Text = InputText
  - UIInput + Secret = InputSecret
  - UISelectOne + Listbox = SelectOneListbox
  - UISelectOne + Menu = SelectOneMenu
  - UISelectOne + Radio = SelectOneRadio
  - UISelectMany + Listbox = SelectManyListbox
  - UISelectMany + Menu = SelectManyMenu
  - UISelectMany + Radio = SelectManyRadio

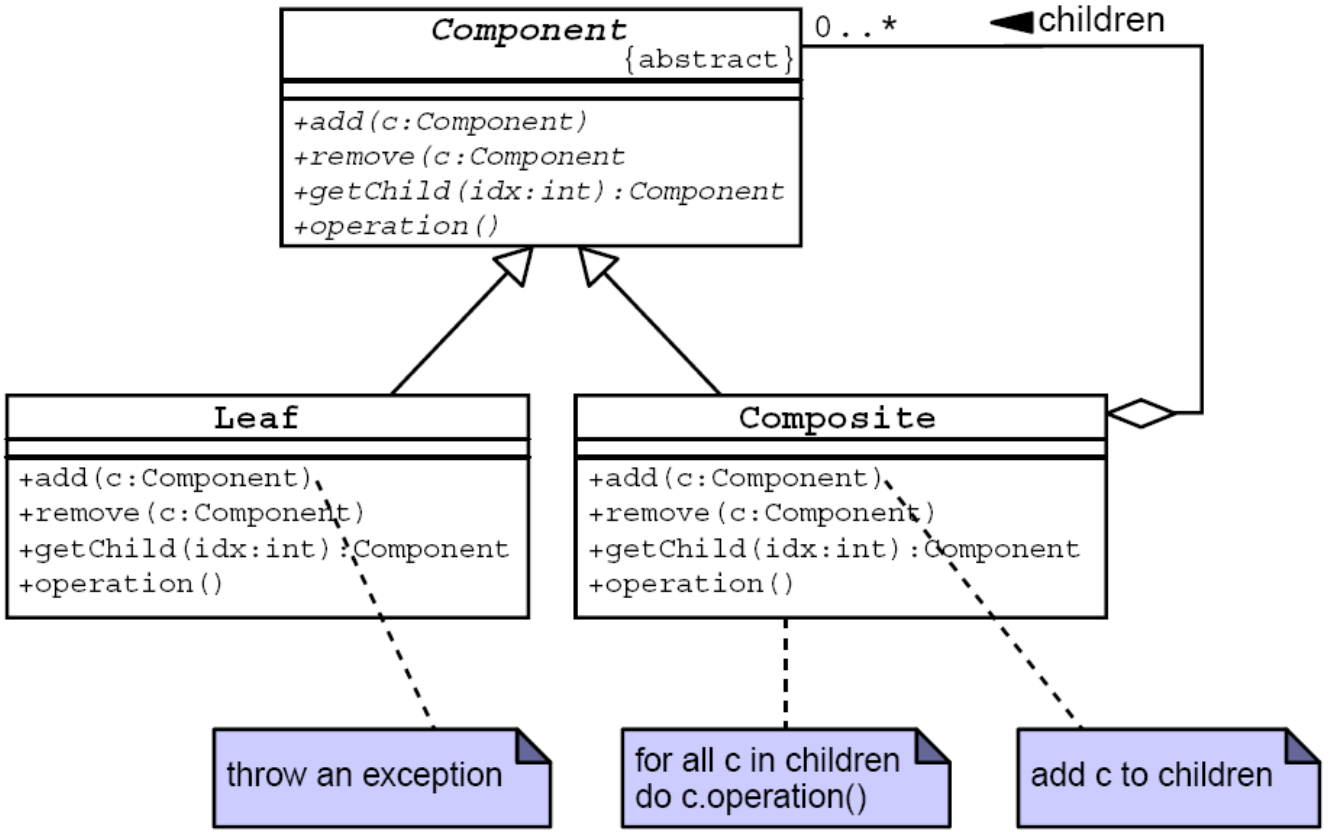
JSF komponens fa

A JavaServer Faces a nézetet (view) komponensek fa struktúrájaként ábrázolja (csakúgy, mint a Swing)



JSF komponens fa (folytatás)

Composite design pattern:



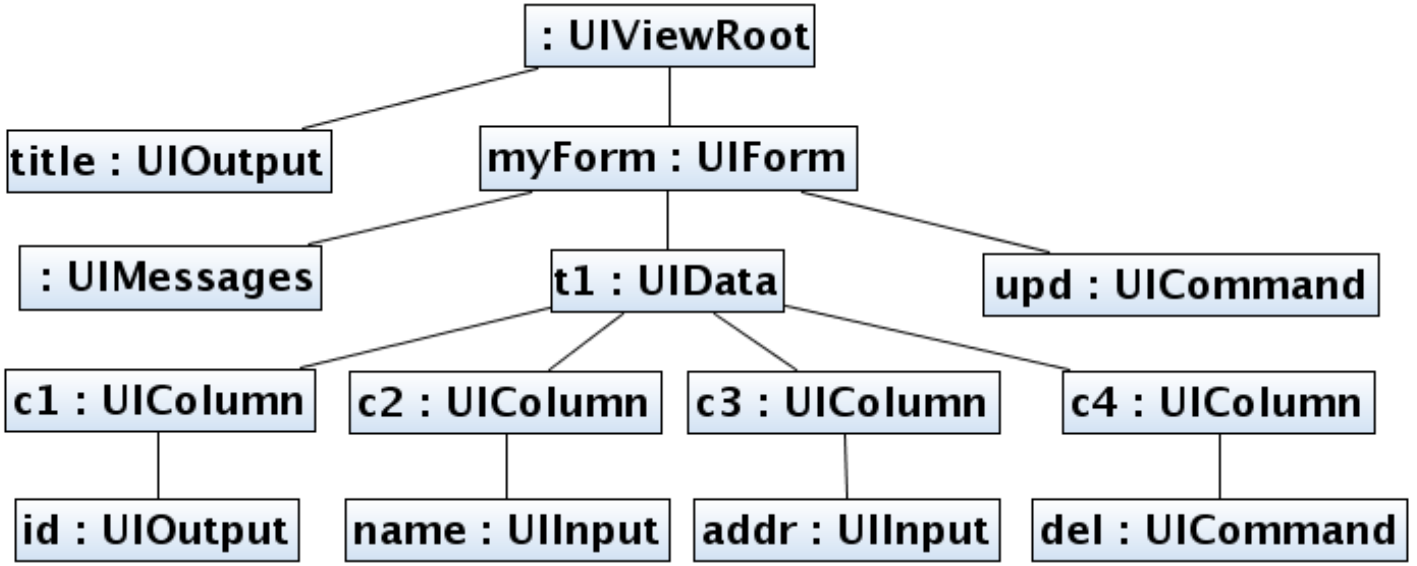
Egy kis példa

List of Customers (8)

111-11-1111	Test Customer	2222 Easy Street, West Beach AZ	Delete
999-45-9034	Perumainar	1444 England Lane, Broomfield CO	Delete
999-78-9012	Anthony Orapallo	123 Tea Street, Columbia MD	Delete
999-90-9009	Terri Cubeta	636 Somewhere Rd, Rosslyn VA	Delete
999-90-8765	Bryan Basham	3290 Course Way, Broomfield CO	Delete
999-33-4444	Georgianna DG Meagher	1000 Mother Court, Columbia MD	Delete
999-44-5555	Tom McGinn	1525 Educator Drive, Burlington MA	Delete
123-45-6789	Varga Péter	Závodszky utca	Delete

Update

A példa komponens fája



A példa komponens fa definíciója

```
<f:view>
  <h1><h:outputText id="title"
   value="List of Customers (#{CustomerList.numberOfCustomers})" /></h1>
    <h:form id="myForm">
        <h:messages />
        <h:dataTable id="t1" value="#{CustomerList.customers}"
        var="customer" cellspacing="2" cellpadding="2"
        style="background-color: grey">
            <h:column id="c1">
                <h:outputText id="id" value="#{customer.ssn}"/>
            </h:column>
            <h:column id="c2">
                <h:inputText id="name" value="#{customer.custName}"
  size="20"/>
            </h:column>
```

A példa komponens fa definíciója (folyt.)

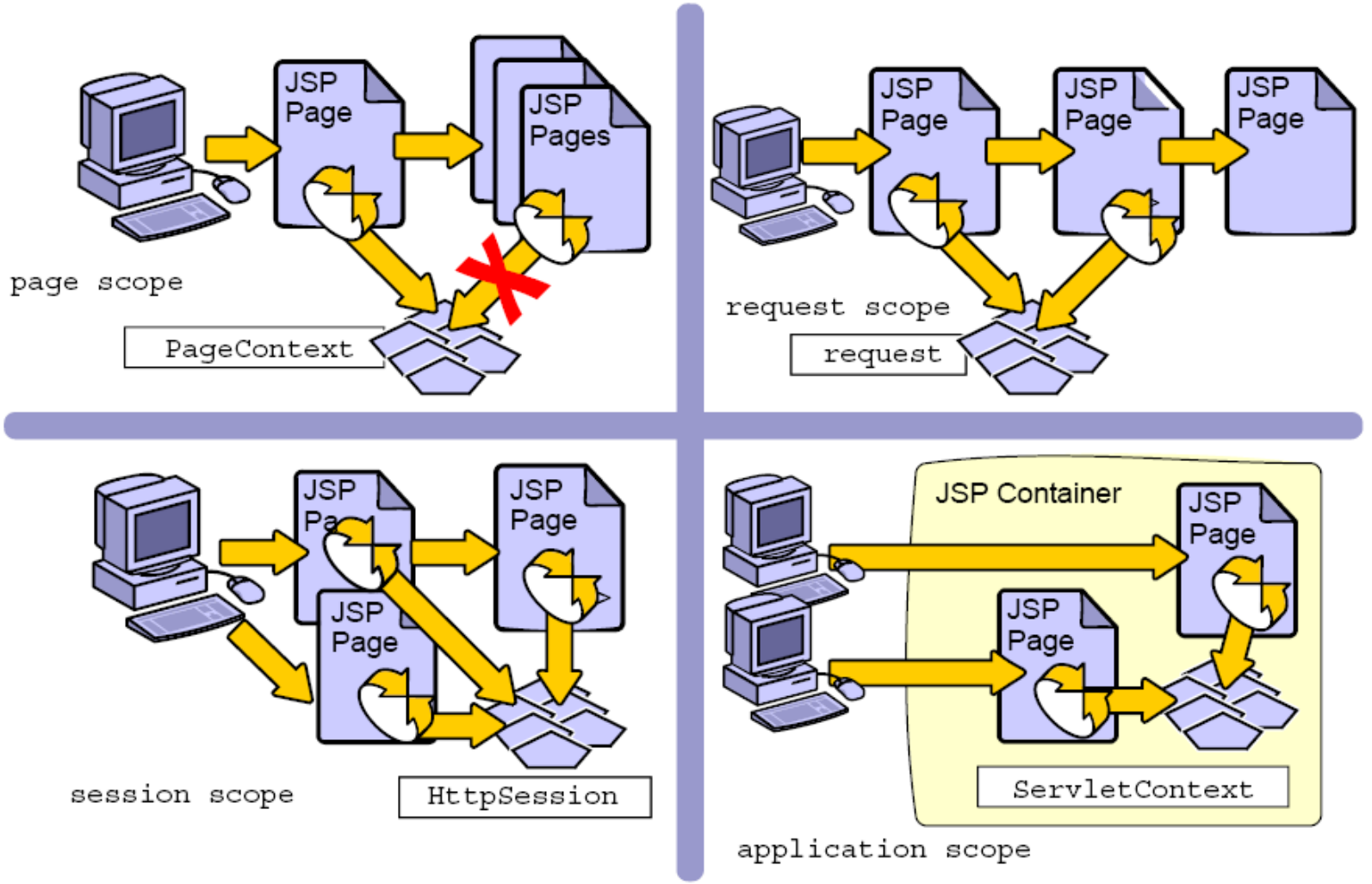
```
            <h:column id="c3">
                <h:inputText id="addr" value="#{customer.address}"
size="40"/>
            </h:column>
            <h:column id="c4">
                <h:commandButton id="del" value="Delete"
action="#{CustomerList.delete}"/>
            </h:column>
        </h:dataTable>

        <h:commandButton id="upd" value="Update"
action="#{CustomerList.update}"/>
    </h:form>
</f:view>
```

Managed bean-ek

- MVC modell-jének reprezentálói
- JavaBeanek, meghatározott névvel, és scope-pal

```
<managed-bean>
  <managed-bean-name>CustomerList</managed-bean-name>
  <managed-bean-class>
    list.CustomerList
  </managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
</managed-bean>
```

Webalkalmazás scope-ok



Komponensek Modell kapcsolata

- Expression Language segítségével
- A managed bean-ek halmaza a kiindulási pont

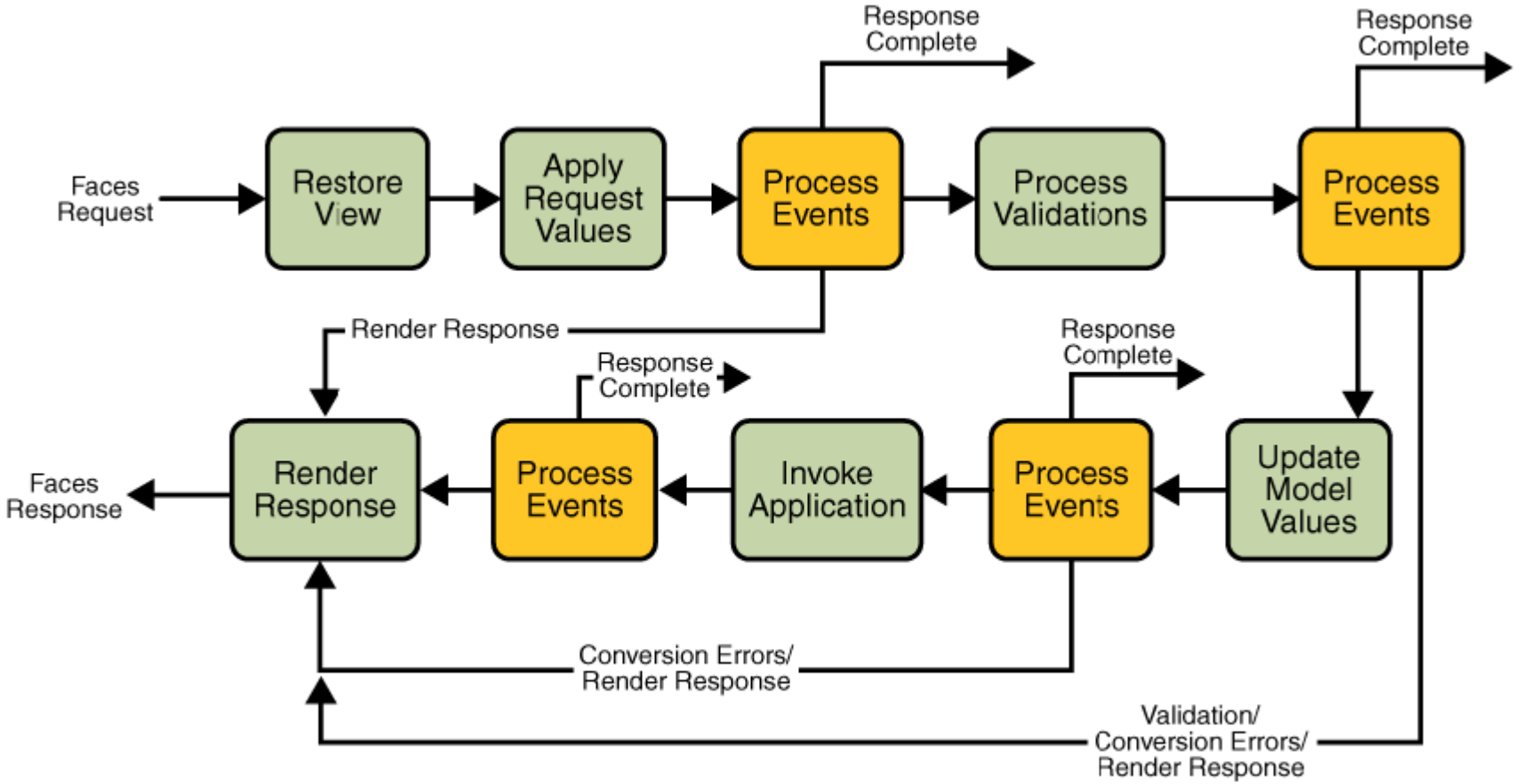
```
<managed-bean>
  <managed-bean-name>CustomerList</managed-bean-name>
  <managed-bean-class>
    list.CustomerList
  </managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
</managed-bean>
```

Komponensek Modell kapcsolata

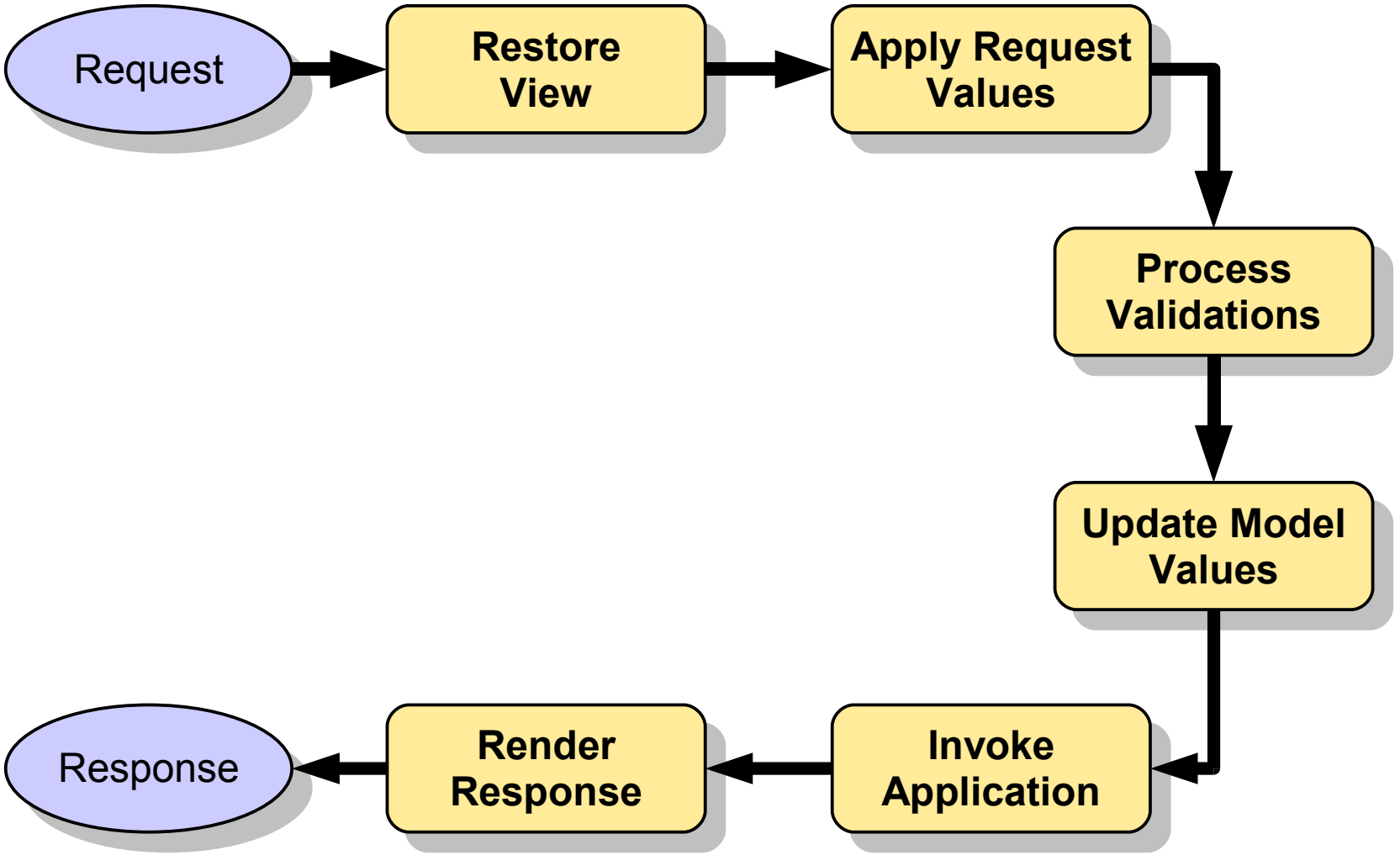
- Expression Language segítségével
- A managed bean-ek halmaza a kiindulási pont

```
<h:dataTable
    id="t1" value="#{CustomerList.customers}"
    var="customer" cellspacing="2" cellpadding="2"
    style="background-color: grey">
```

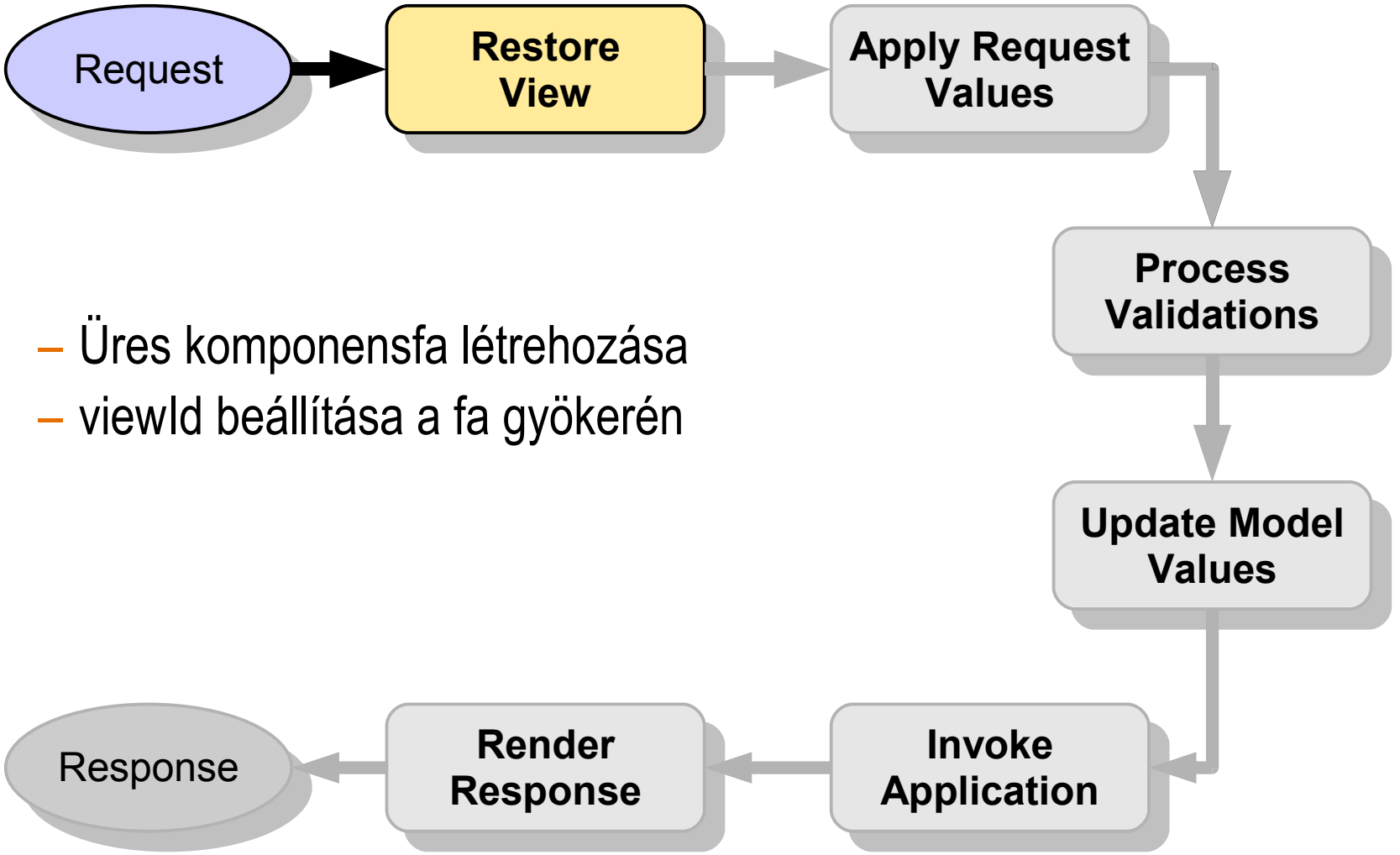
A JSF kérésfeldolgozási ciklusa



A JSF kérésfeldolgozási ciklusa

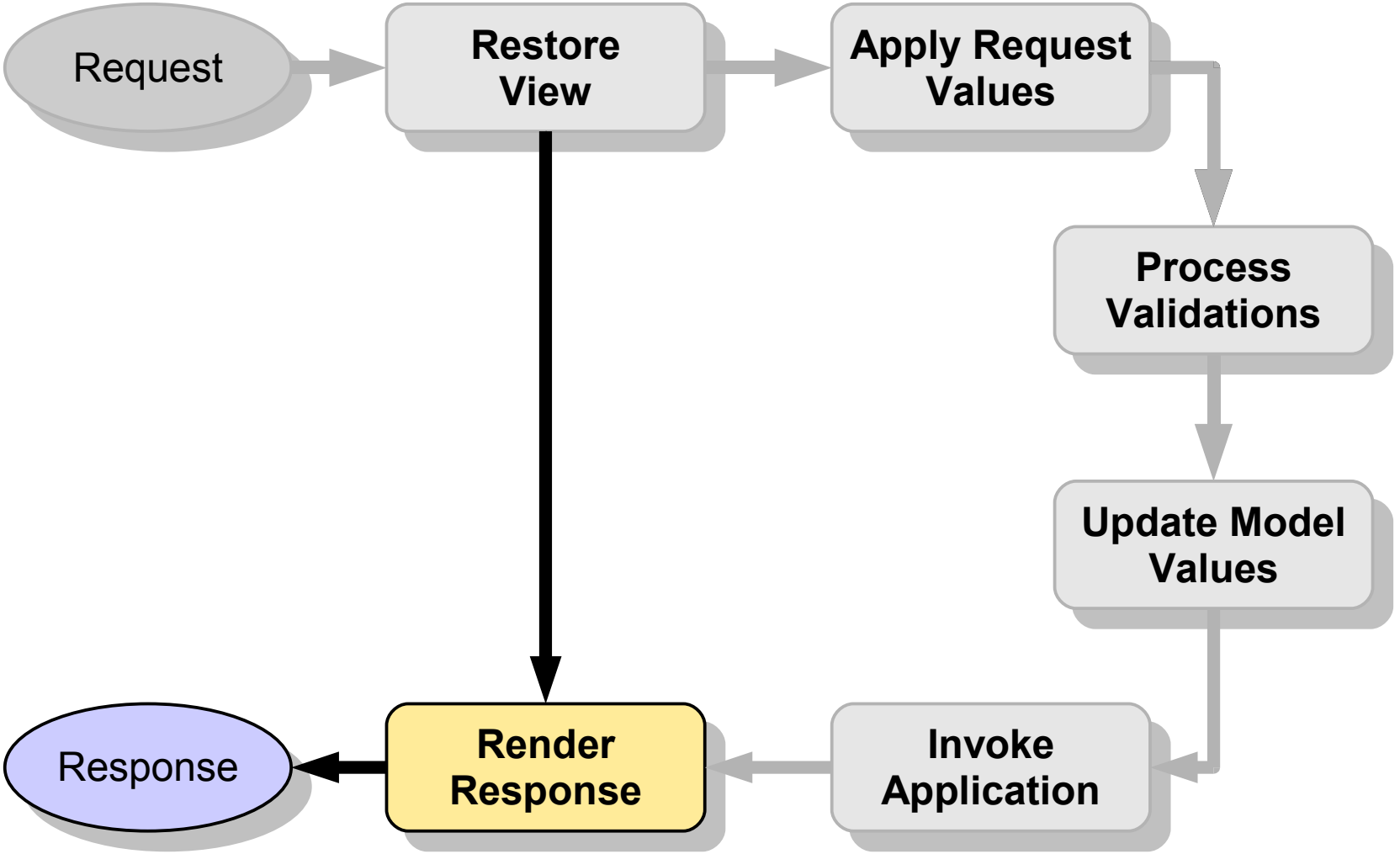


A JSF kérésfeldolgozás – első lekérés



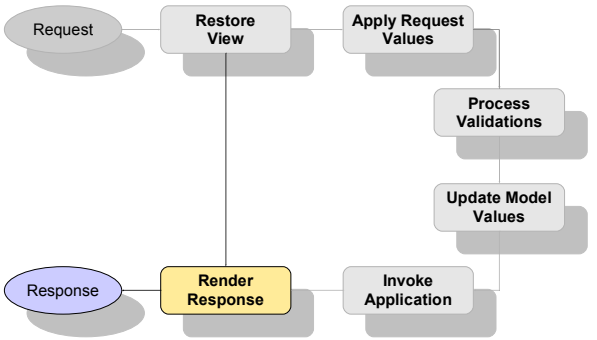
- Üres komponensfa létrehozása
- viewId beállítása a fa gyökerén

A JSF kérésfeldolgozás – első lekérés

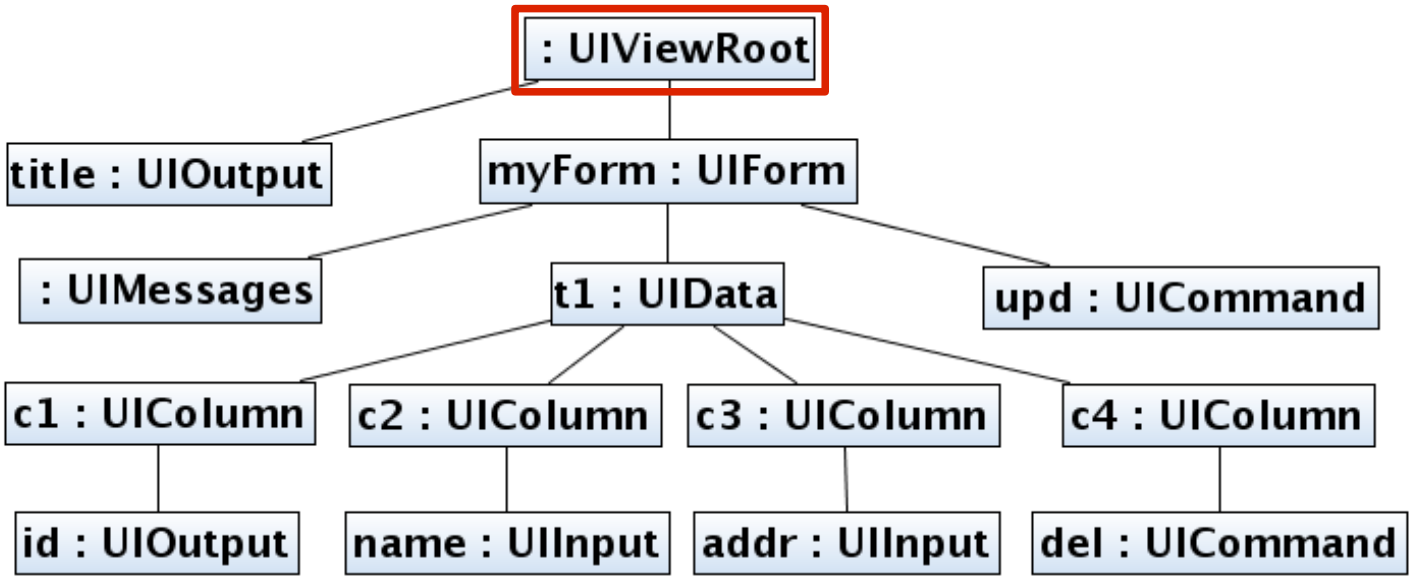


A Render Response fázis teendői

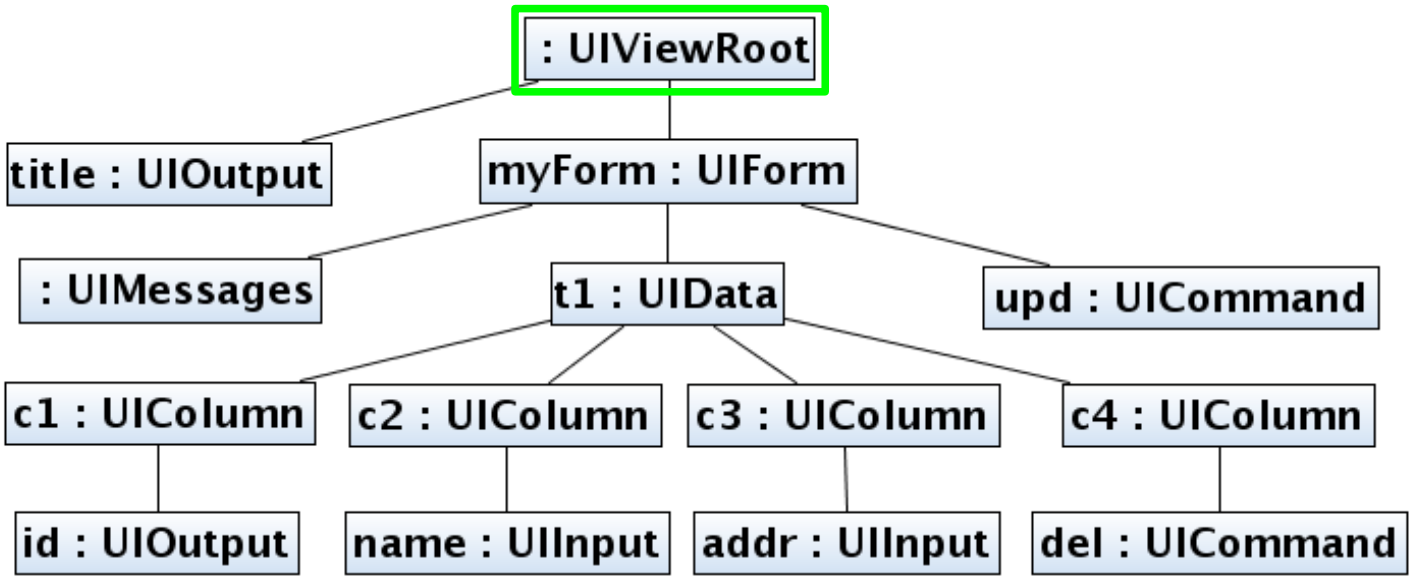
- Fabejárás három fázisban:
  - encodeBegin
  - encodeChildren
  - encodeEnd
- A szabványos komponensek a renderer ugyanilyen nevű metódusát hívják



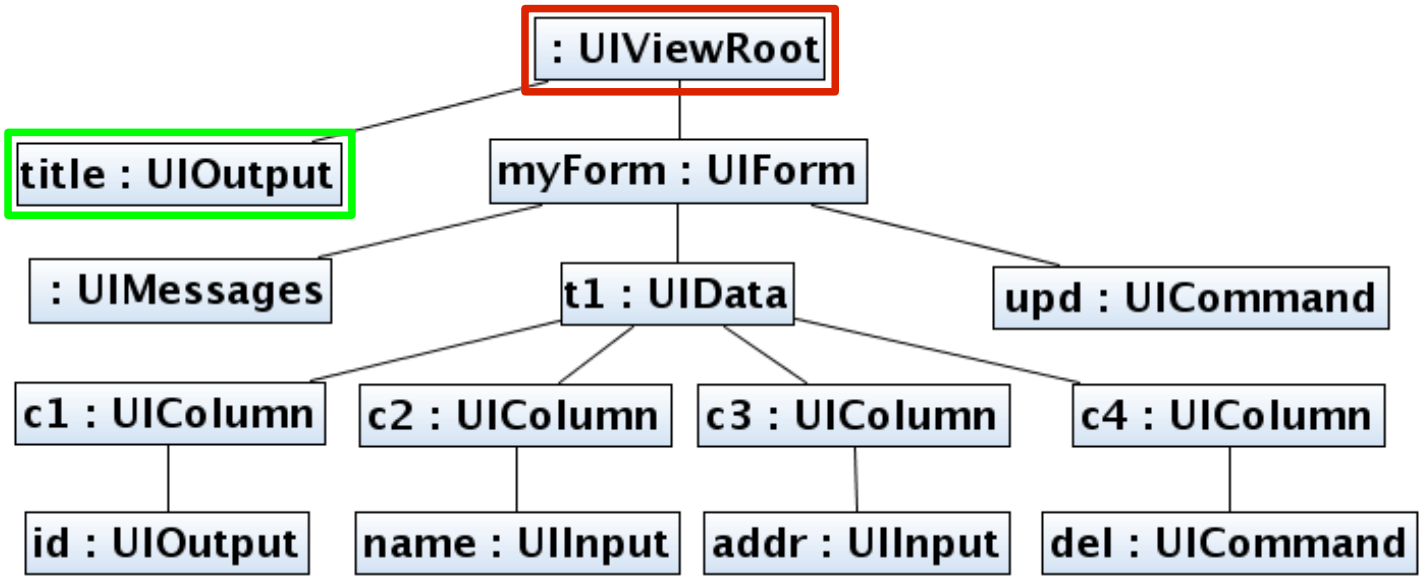
A fabejárás


: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

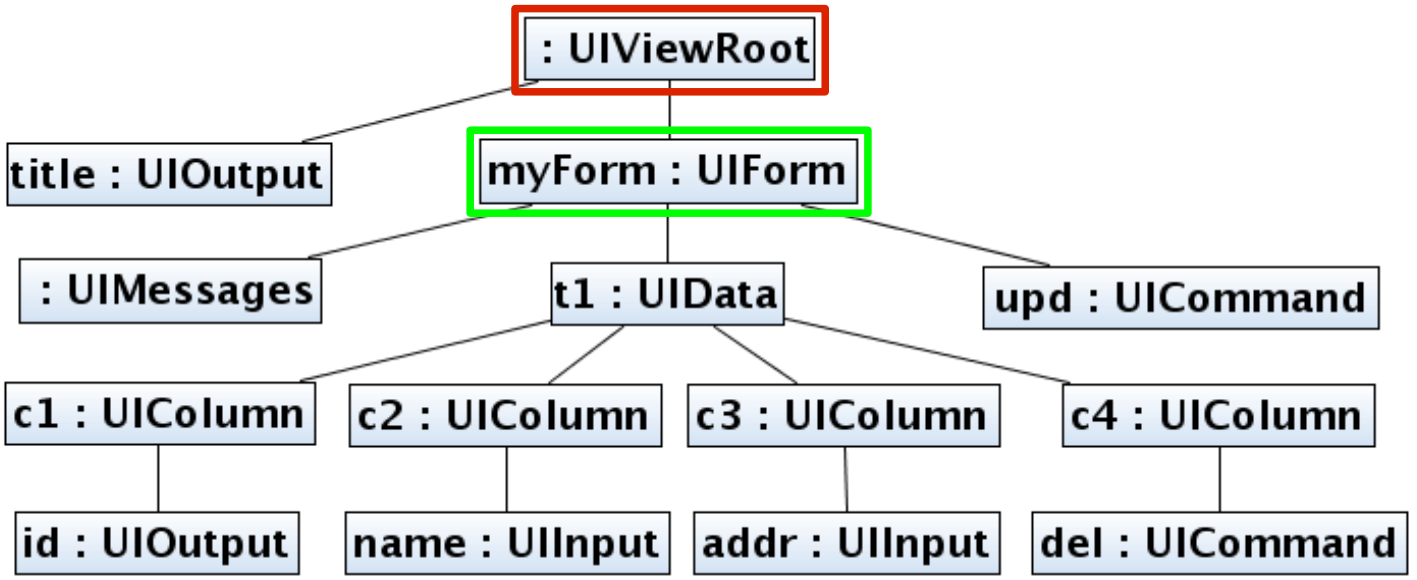
A fabejárás


: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

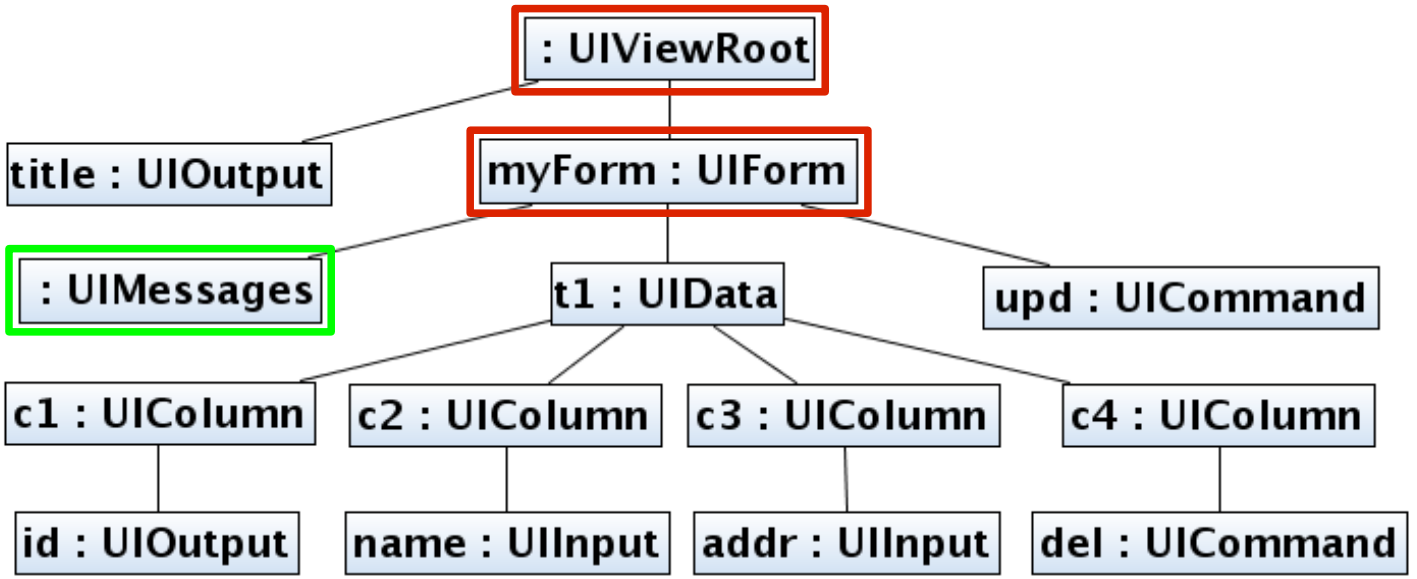
A fabejárás


: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

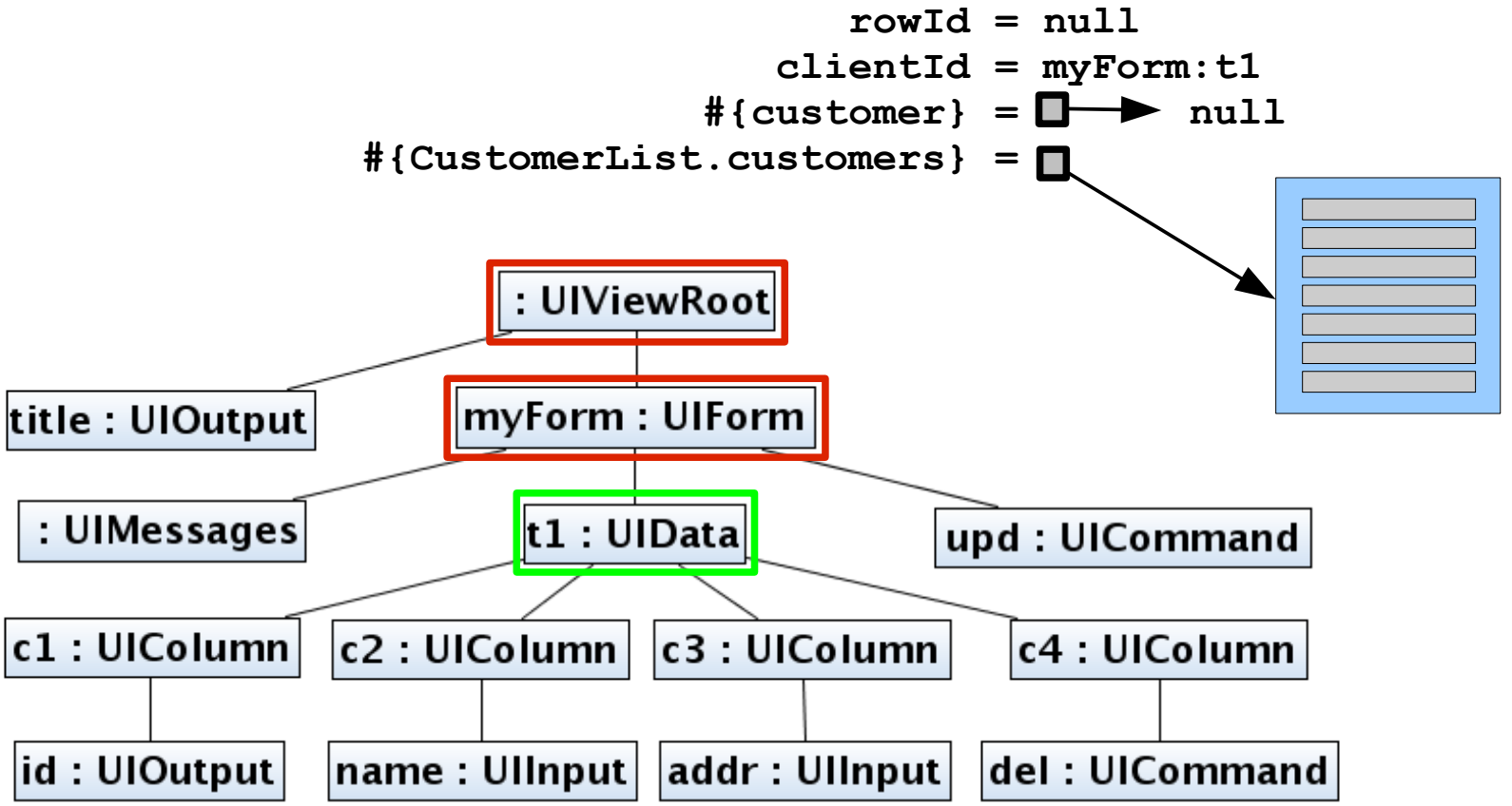
A fabejárás


: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

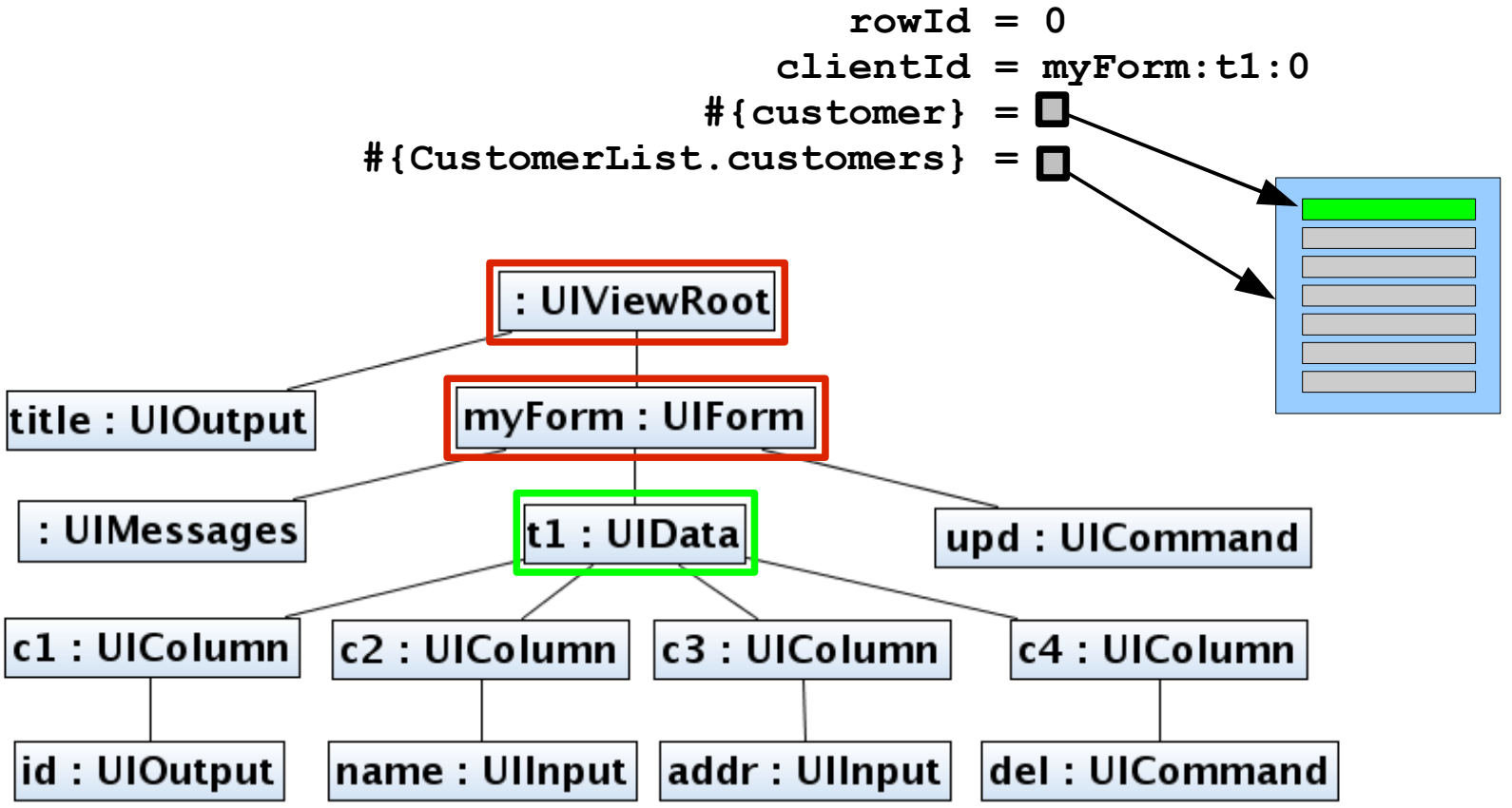
A fabejárás


: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

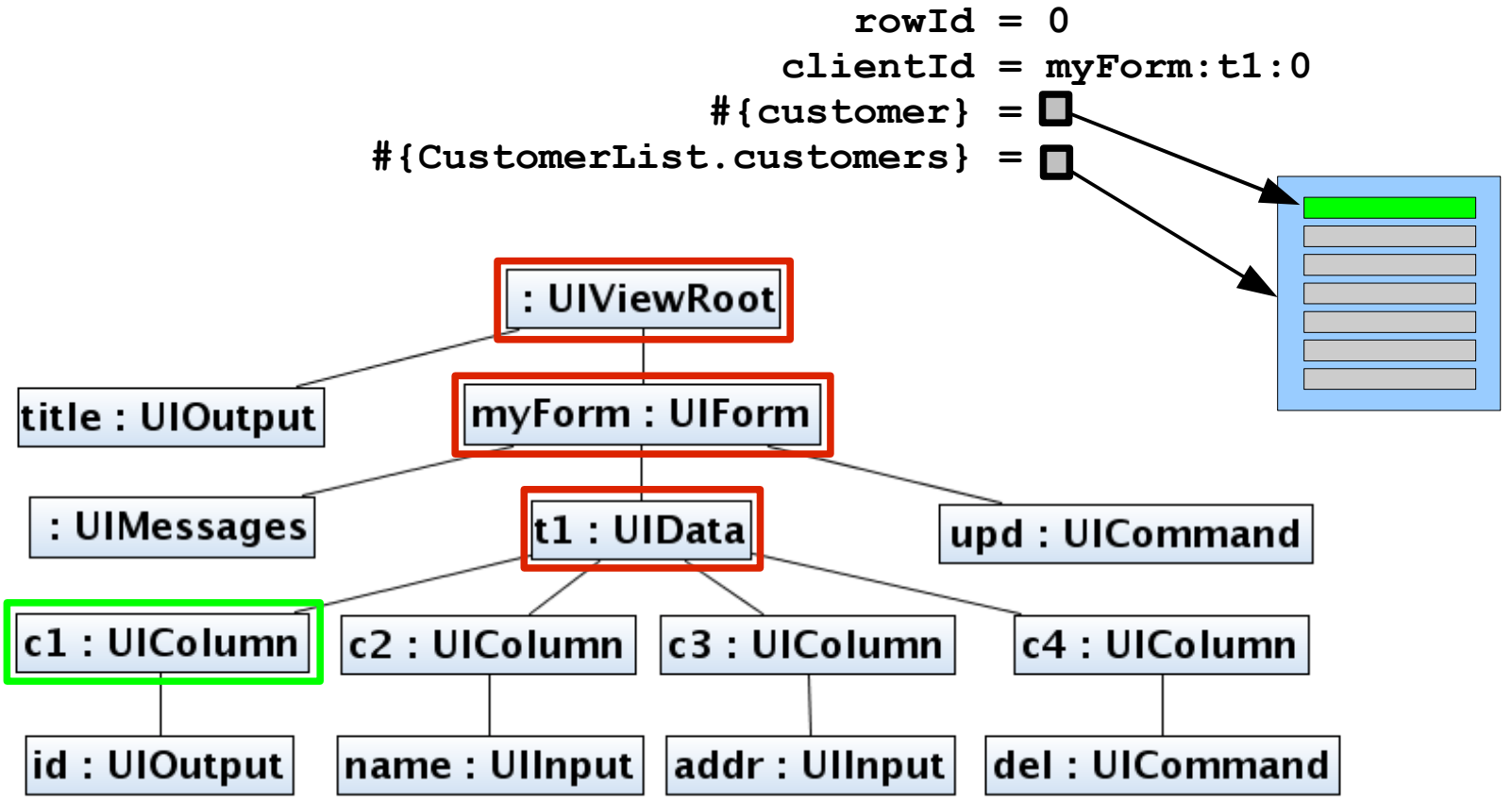
A fabejárás


rowId = null
clientId = myForm:t1
#{customer} = null
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

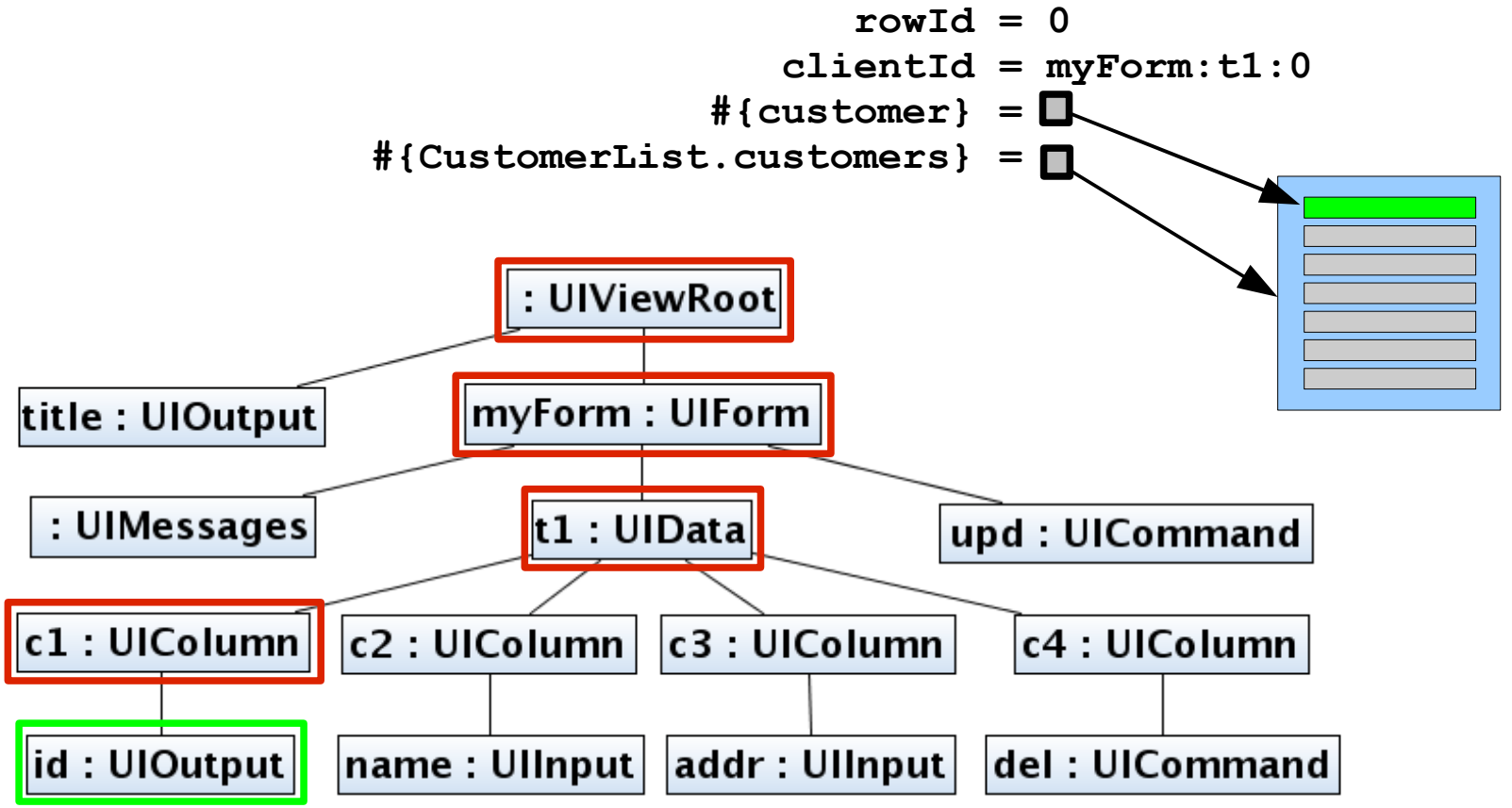
A fabejárás


rowId = 0
clientId = myForm:t1:0
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

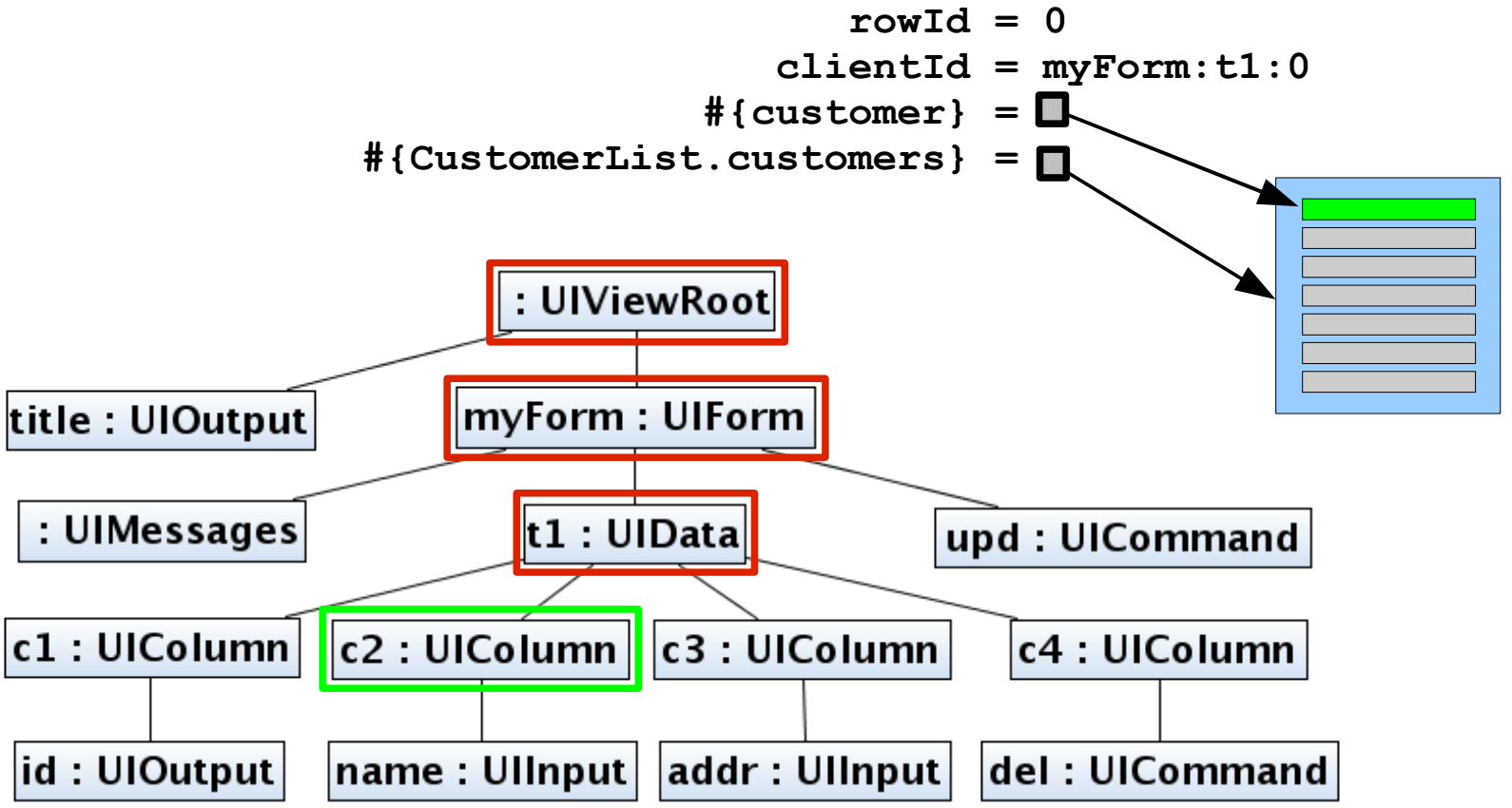
A fabejárás


rowId = 0
clientId = myForm:t1:0
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

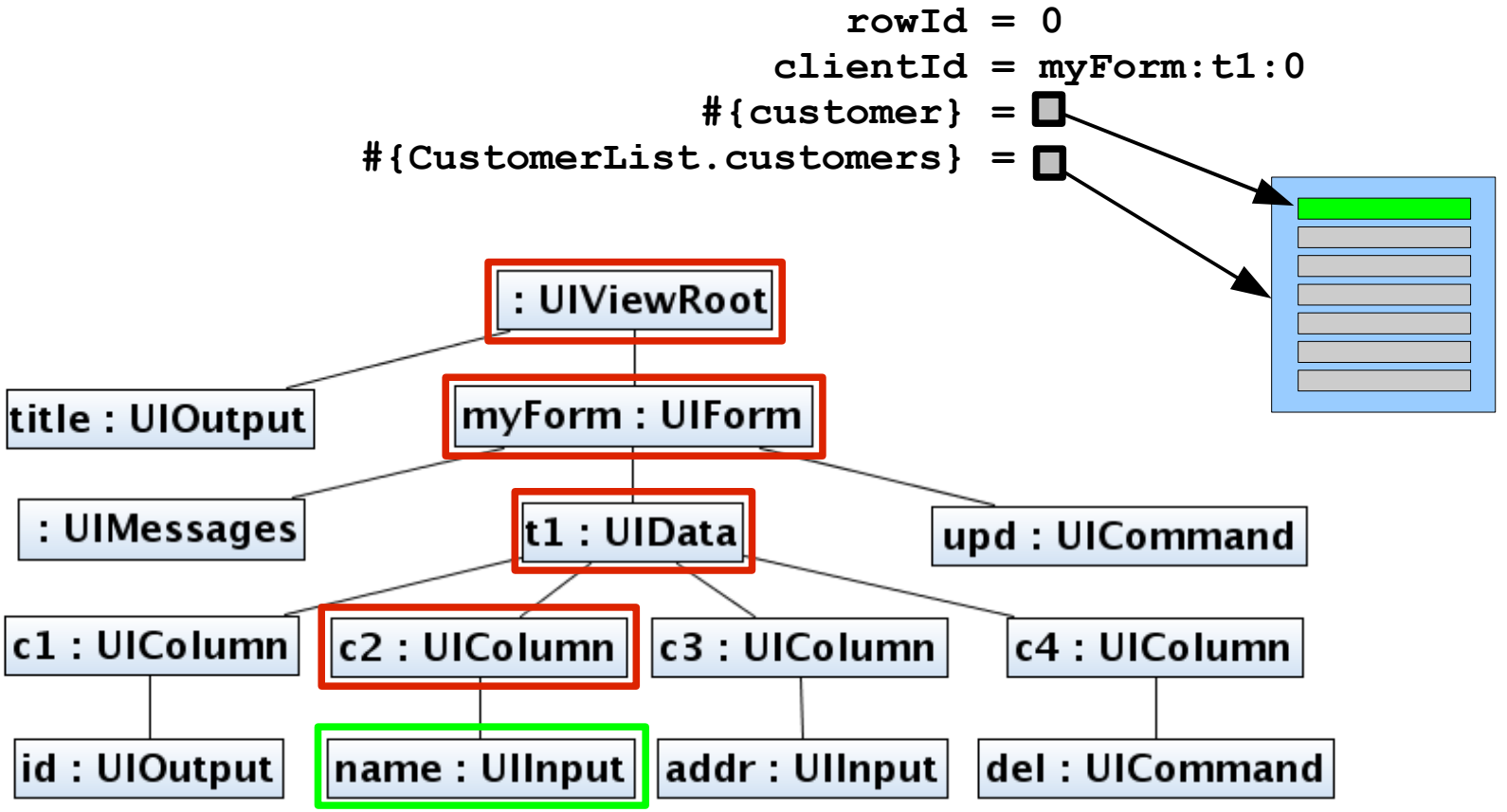
A fabejárás


rowId = 0
clientId = myForm:t1:0
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

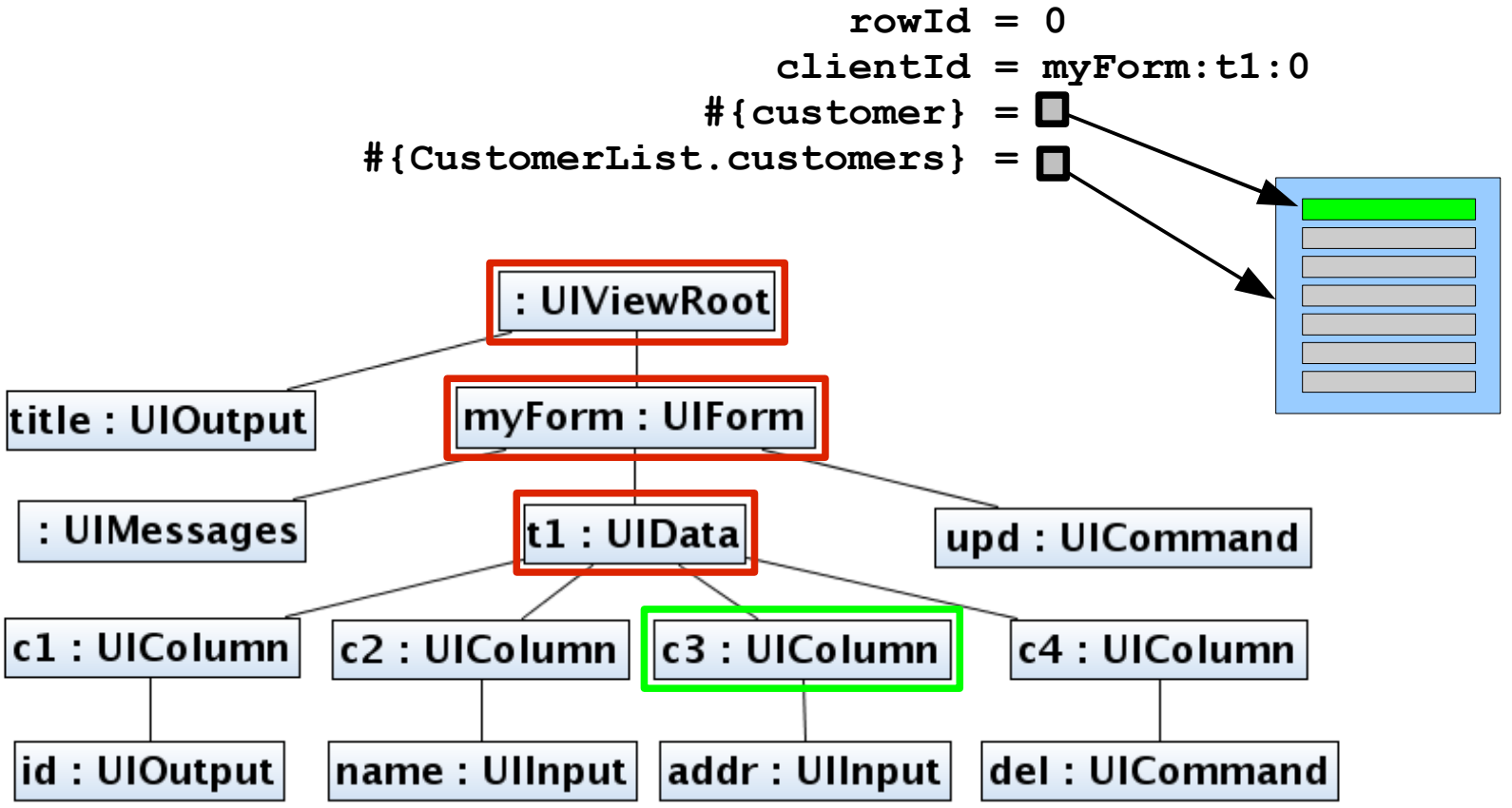
A fabejárás


rowId = 0
clientId = myForm:t1:0
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

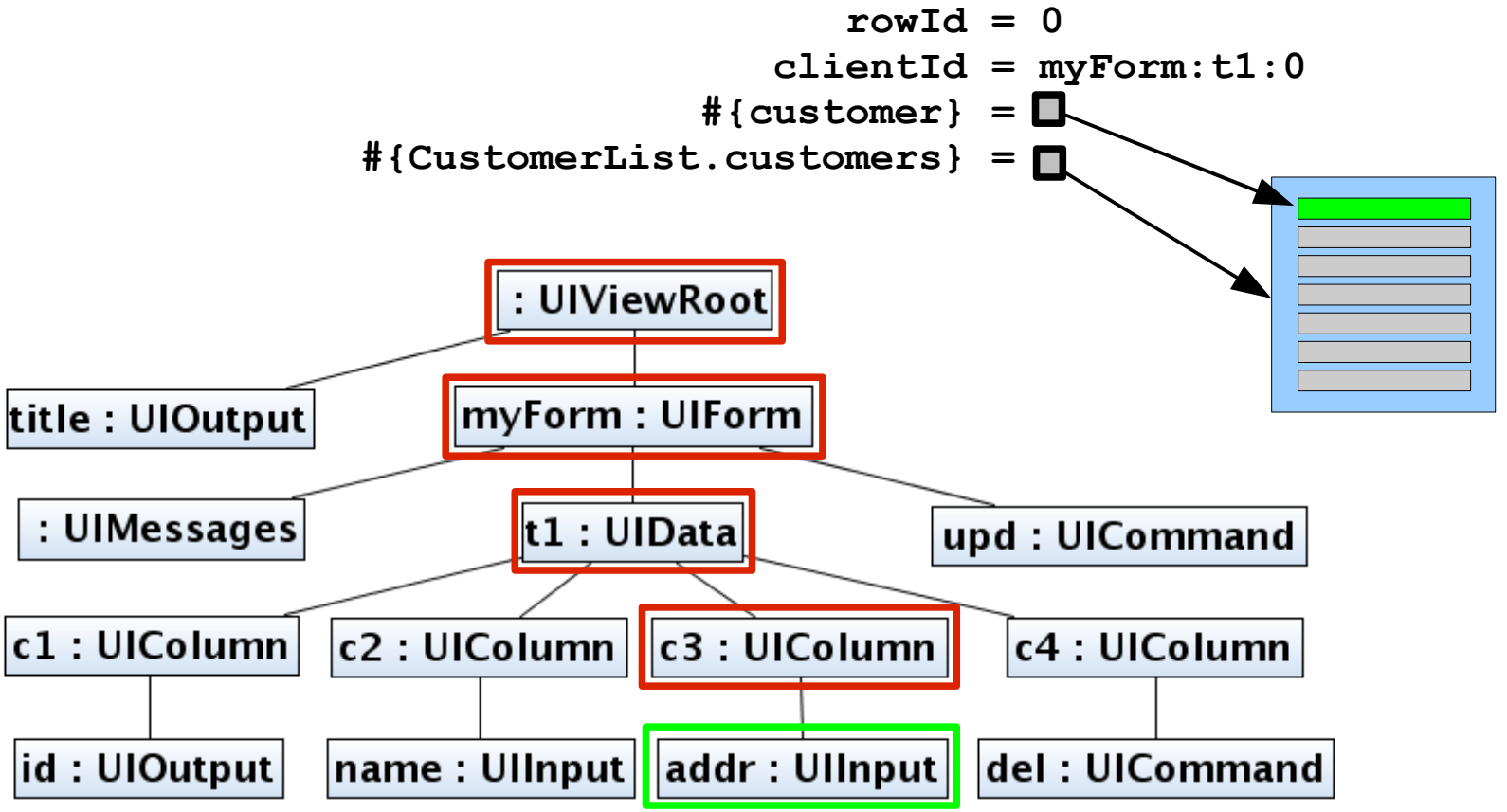
A fabejárás


rowId = 0
clientId = myForm:t1:0
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

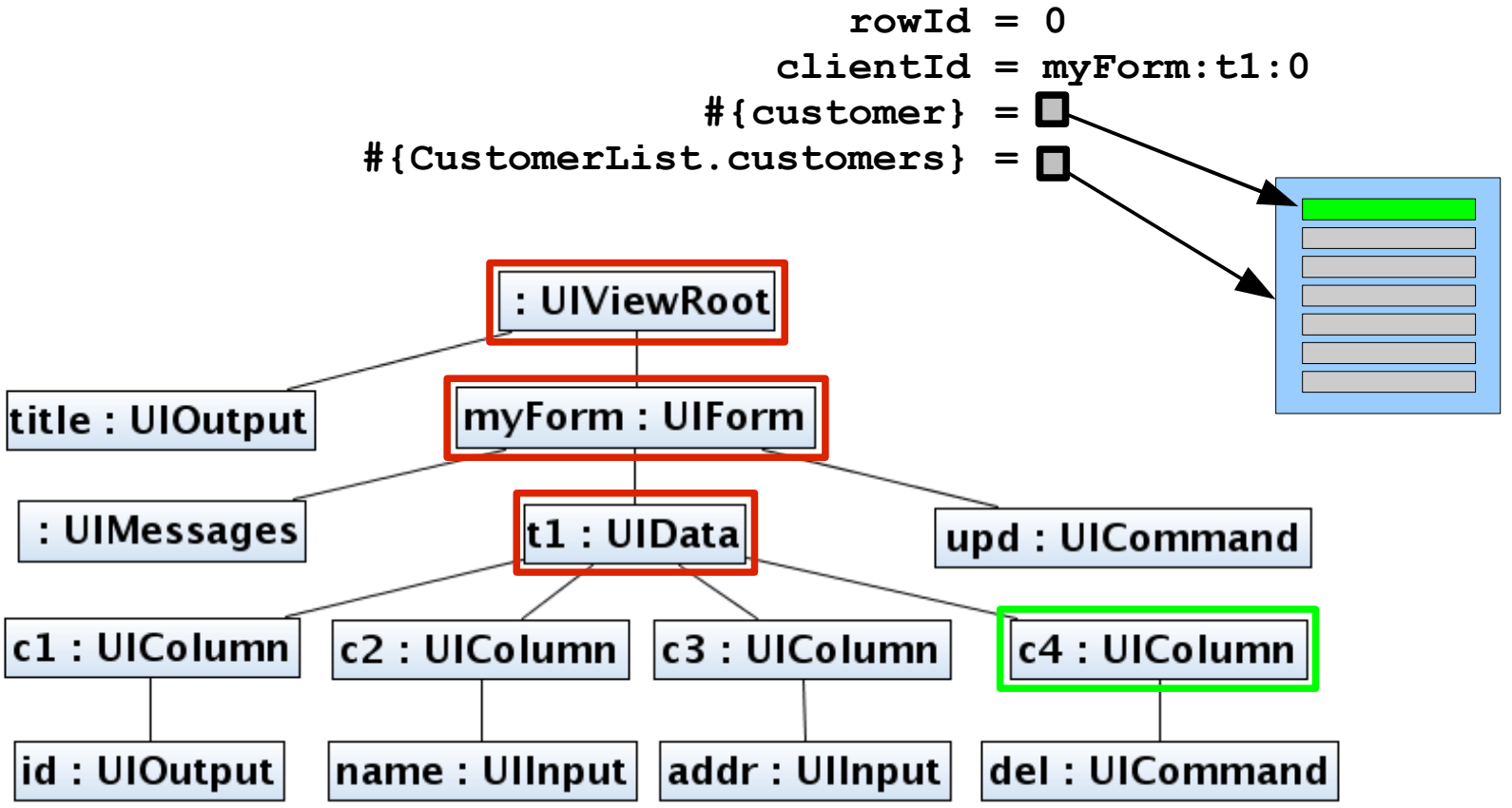
A fabejárás


rowId = 0
clientId = myForm:t1:0
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

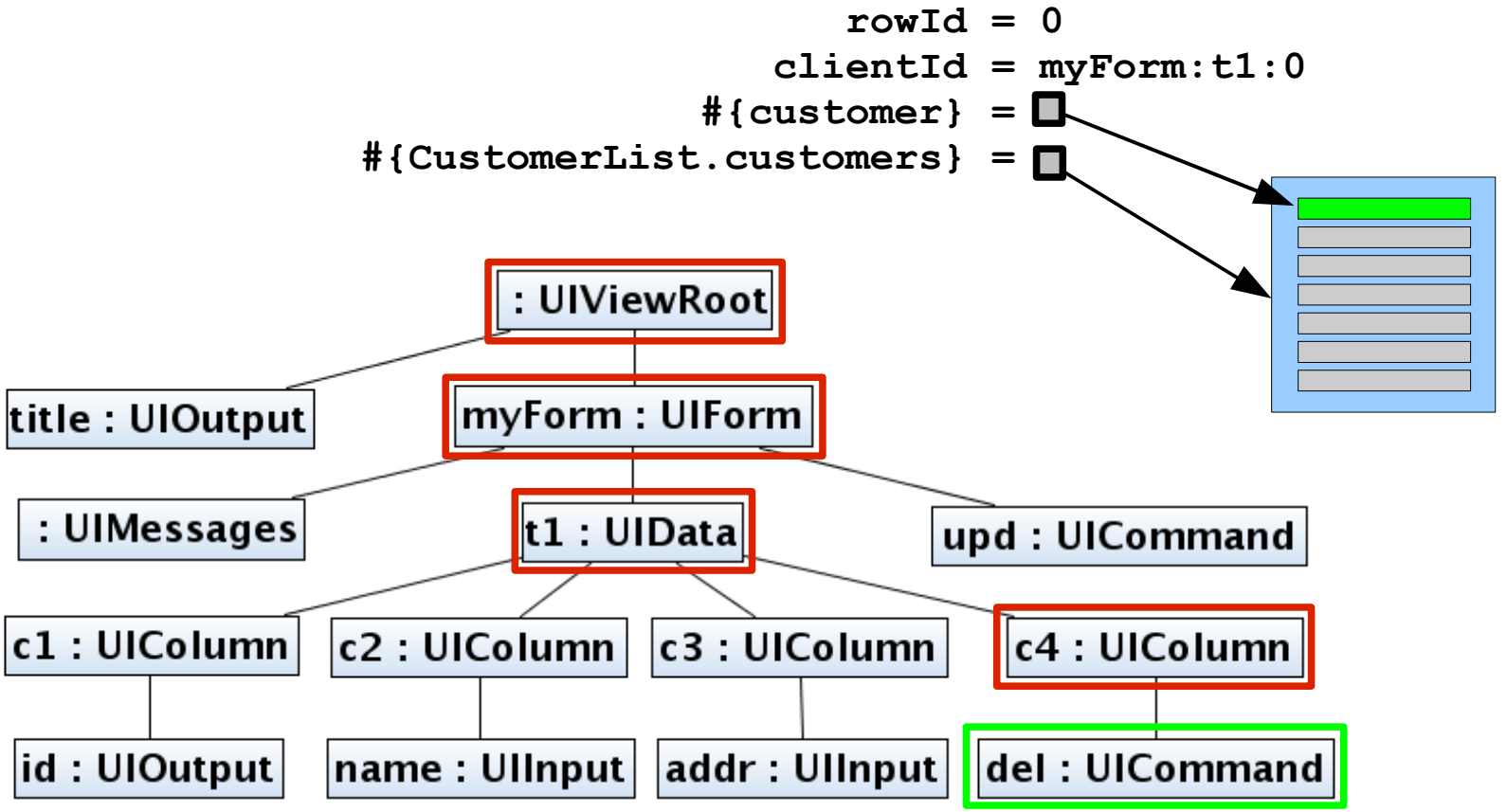
A fabejárás


rowId = 0
clientId = myForm:t1:0
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

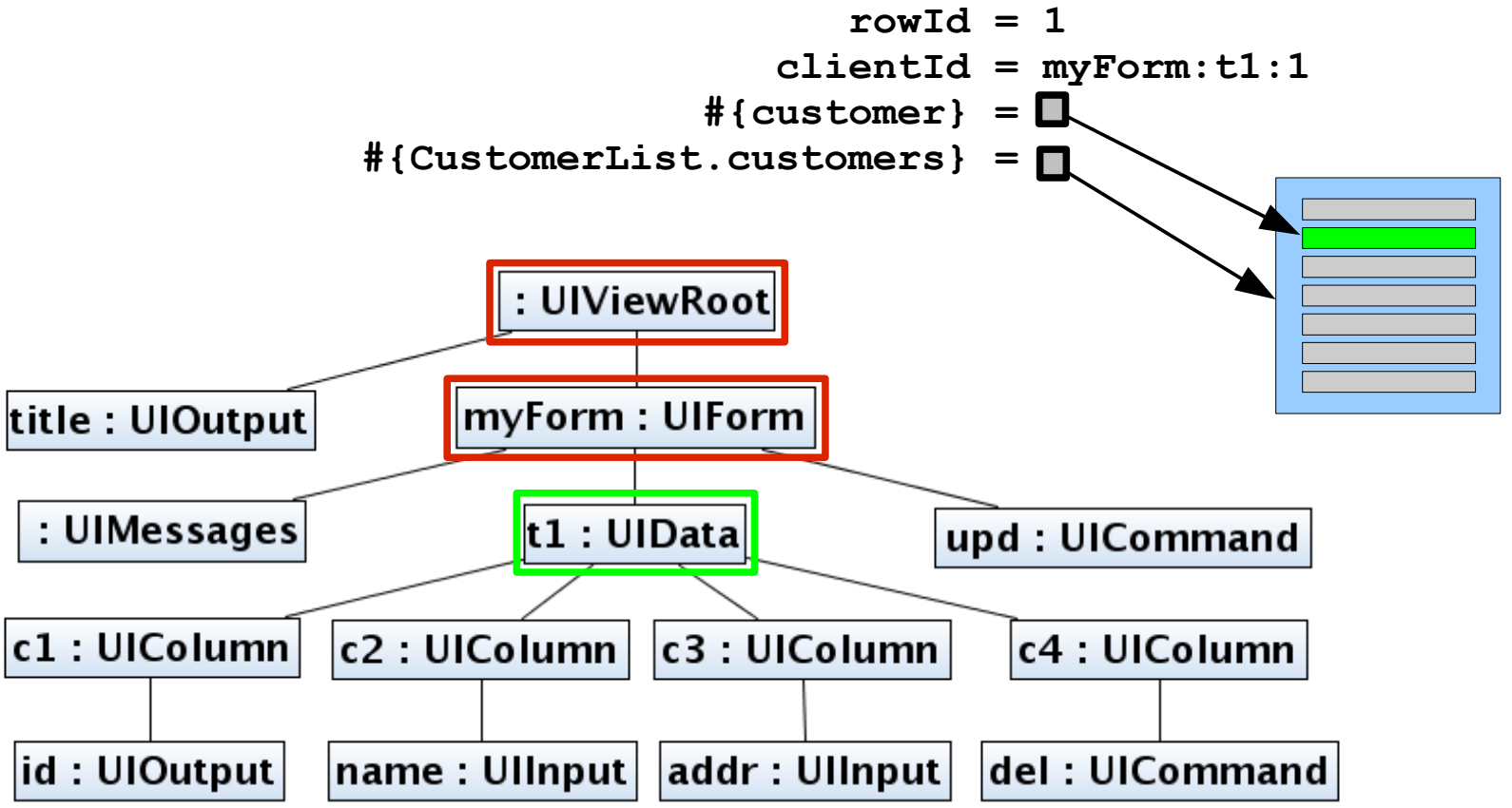
A fabejárás


rowId = 0
clientId = myForm:t1:0
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

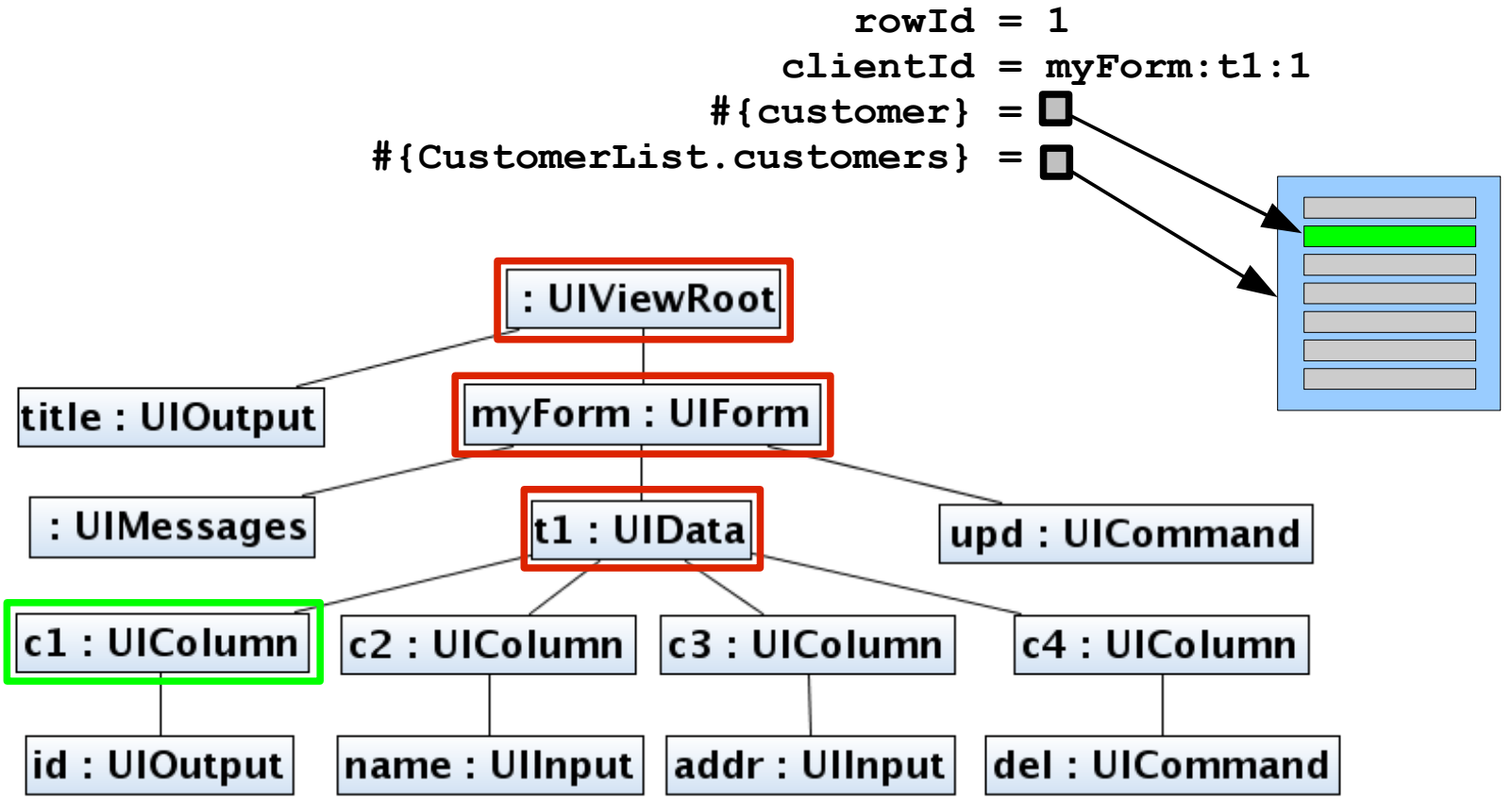
A fabejárás


rowId = 0
clientId = myForm:t1:0
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

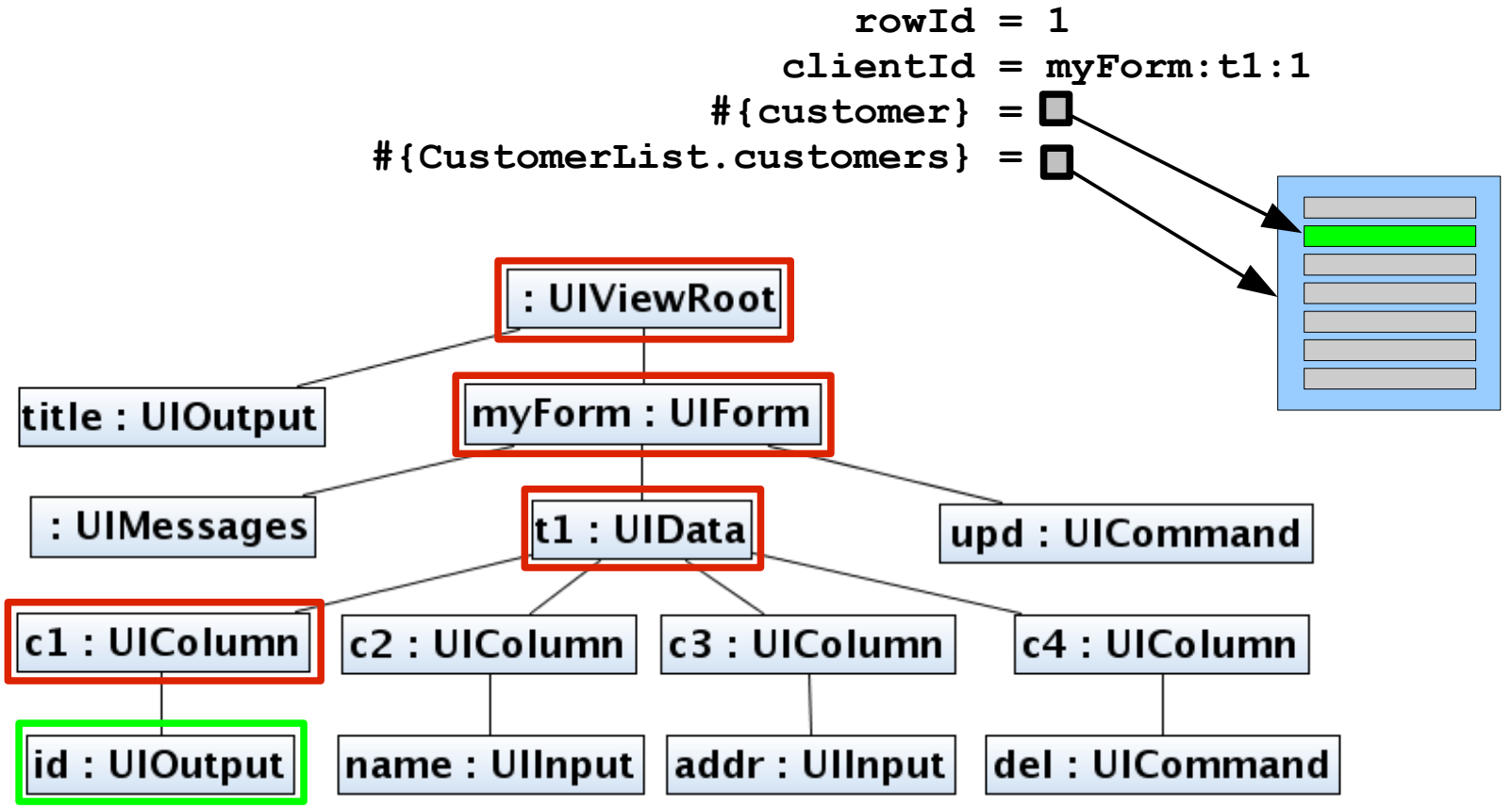
A fabejárás


rowId = 1
clientId = myForm:t1:1
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

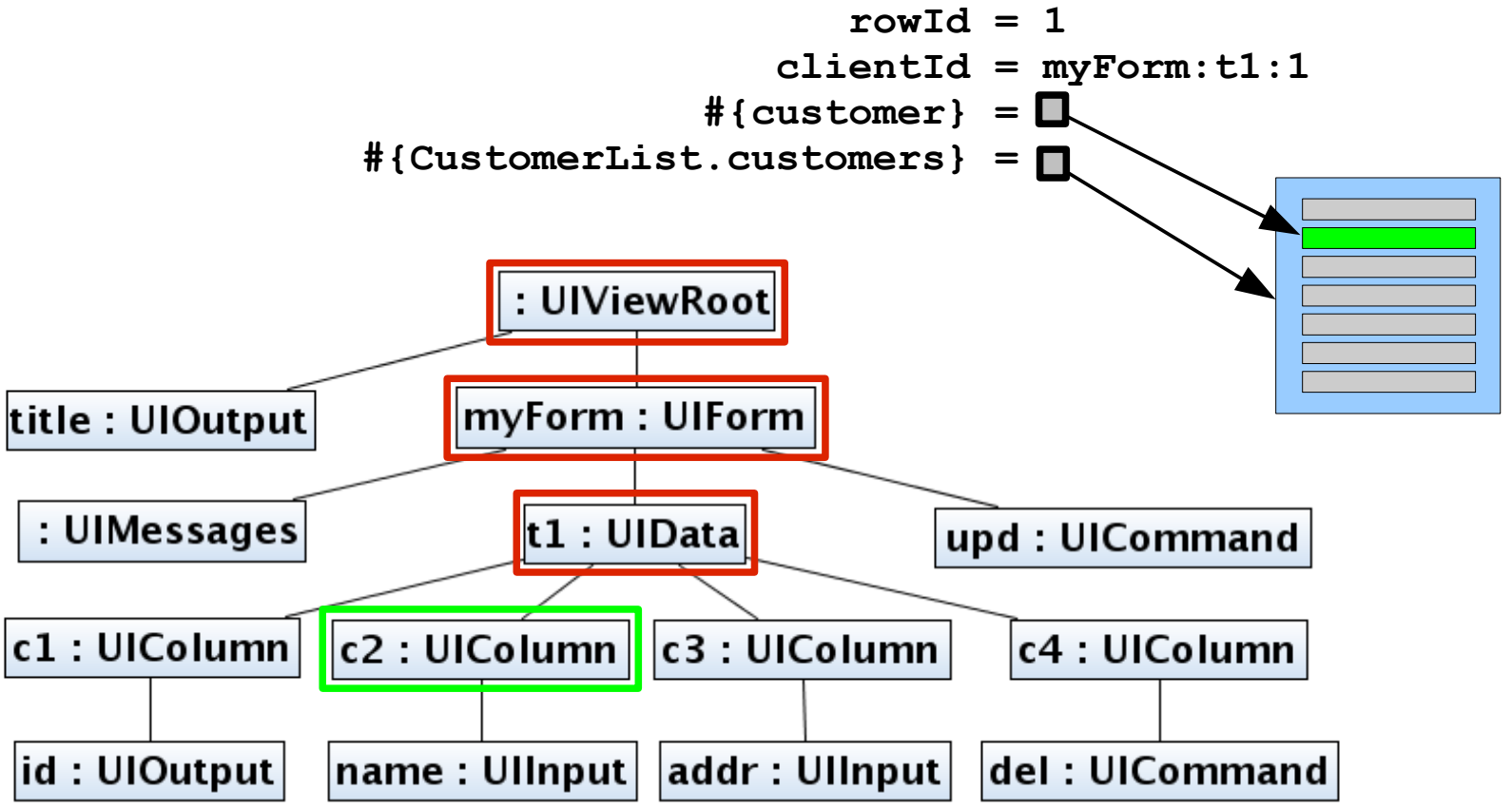
A fabejárás


rowId = 1
clientId = myForm:t1:1
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

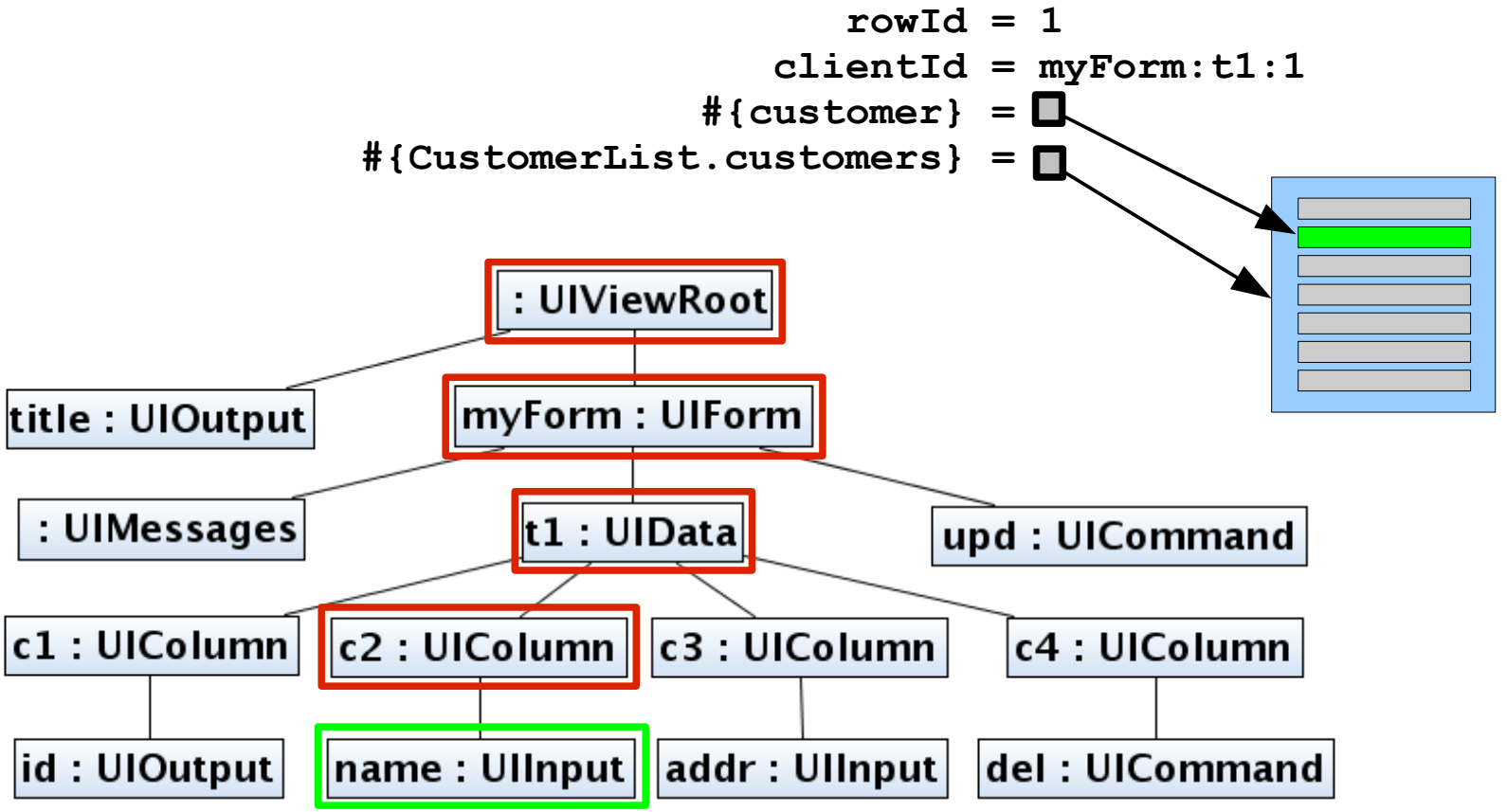
A fabejárás


rowId = 1
clientId = myForm:t1:1
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

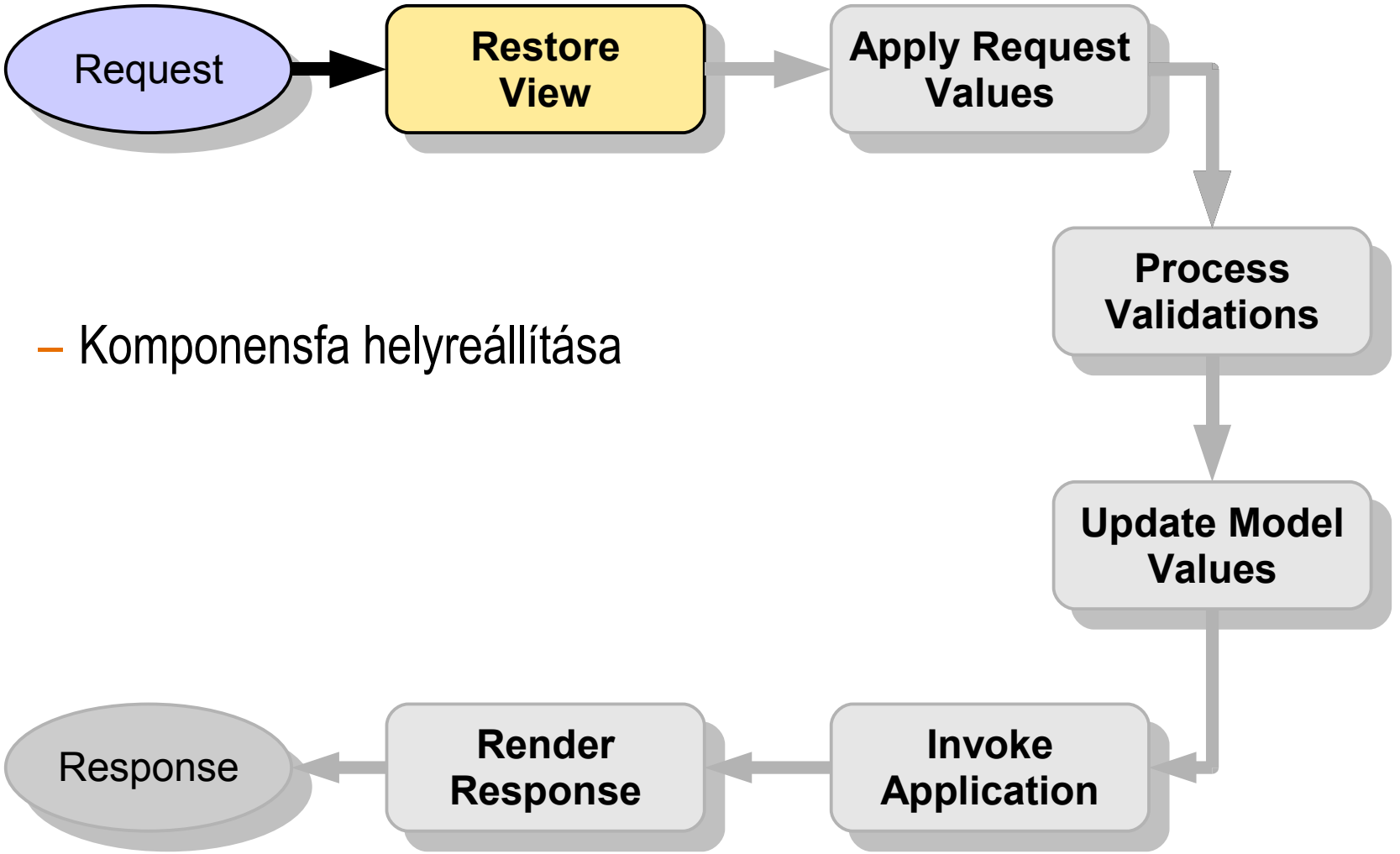
A fabejárás


rowId = 1
clientId = myForm:t1:1
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

A fabejárás

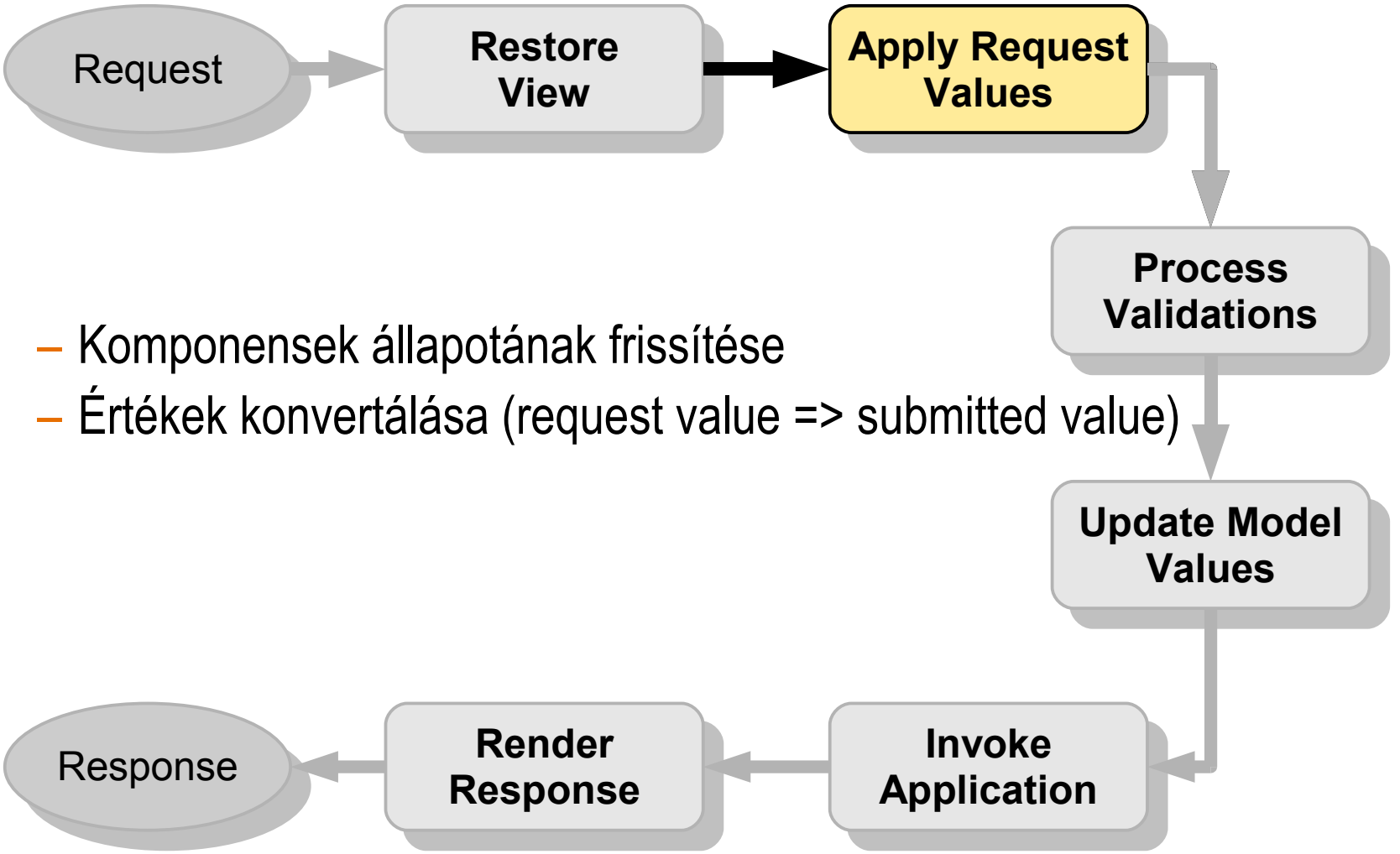

rowId = 1
clientId = myForm:t1:1
#{customer} =
#{CustomerList.customers} =
: UIViewRoot
title : UIOutput
myForm : UIForm
: UIMessages
t1 : UIData
upd : UICommand
c1 : UIColumn
c2 : UIColumn
c3 : UIColumn
c4 : UIColumn
id : UIOutput
name : UIInput
addr : UIInput
del : UICommand

A JSF kérésfeldolgozás – visszaküldés



- Komponensfa helyreállítása

A JSF kérésfeldolgozás – visszaküldés



- Komponensek állapotának frissítése
- Értékek konvertálása (request value => submitted value)

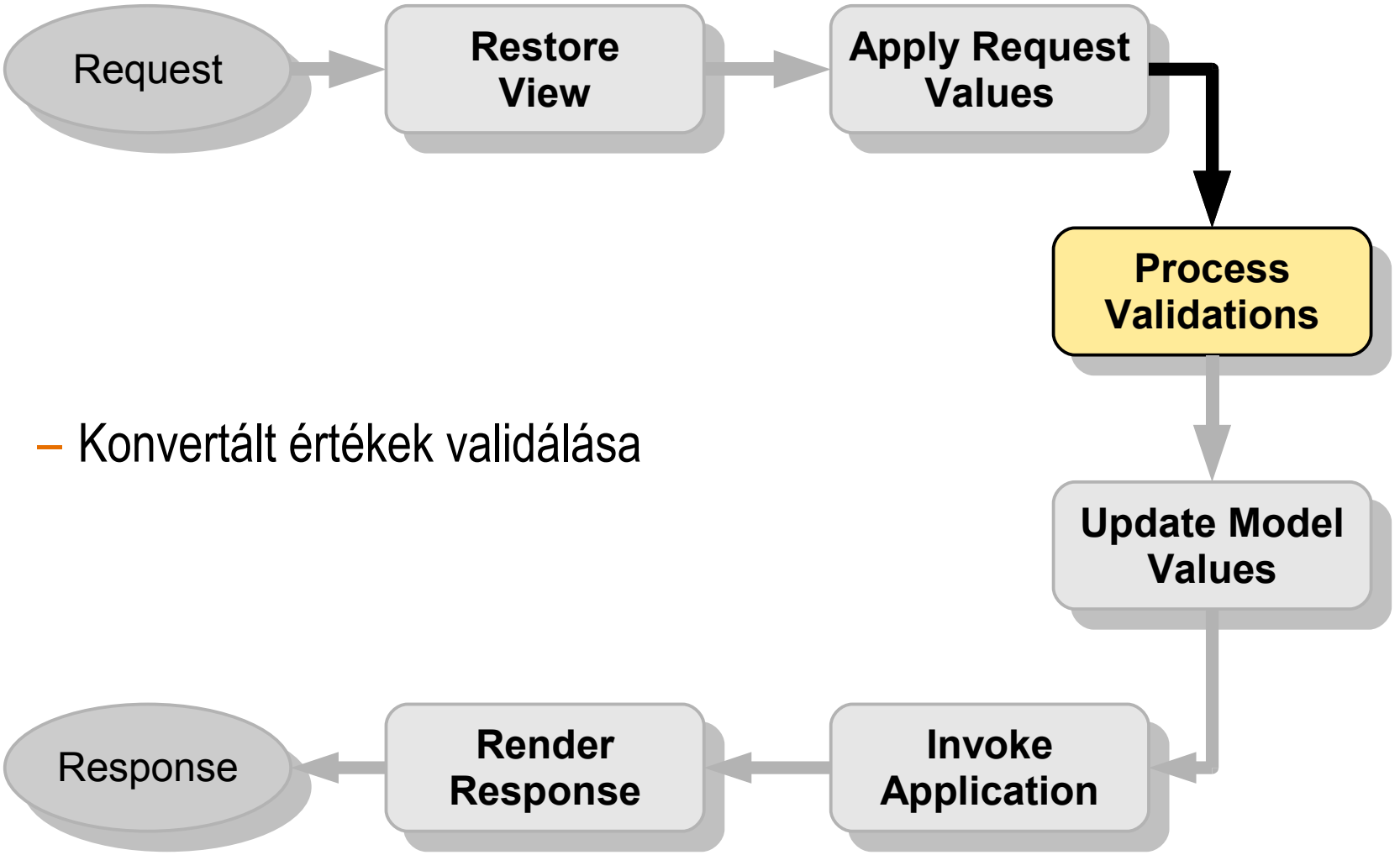
A konvertálási mechanizmus

- Egyszerű interfész (javax.faces.convert.Converter):
  - `getAsObject(String) : Object`
  - `getAsString(Object) : String`
- Számos beépített konverter:
  - Primitív típusokra
  - DateTimeConverter
  - EnumConverter
- Converter hozzárendelési módok
  - converterID alapján
  - Típus (Class) alapján

Eseménykezelés

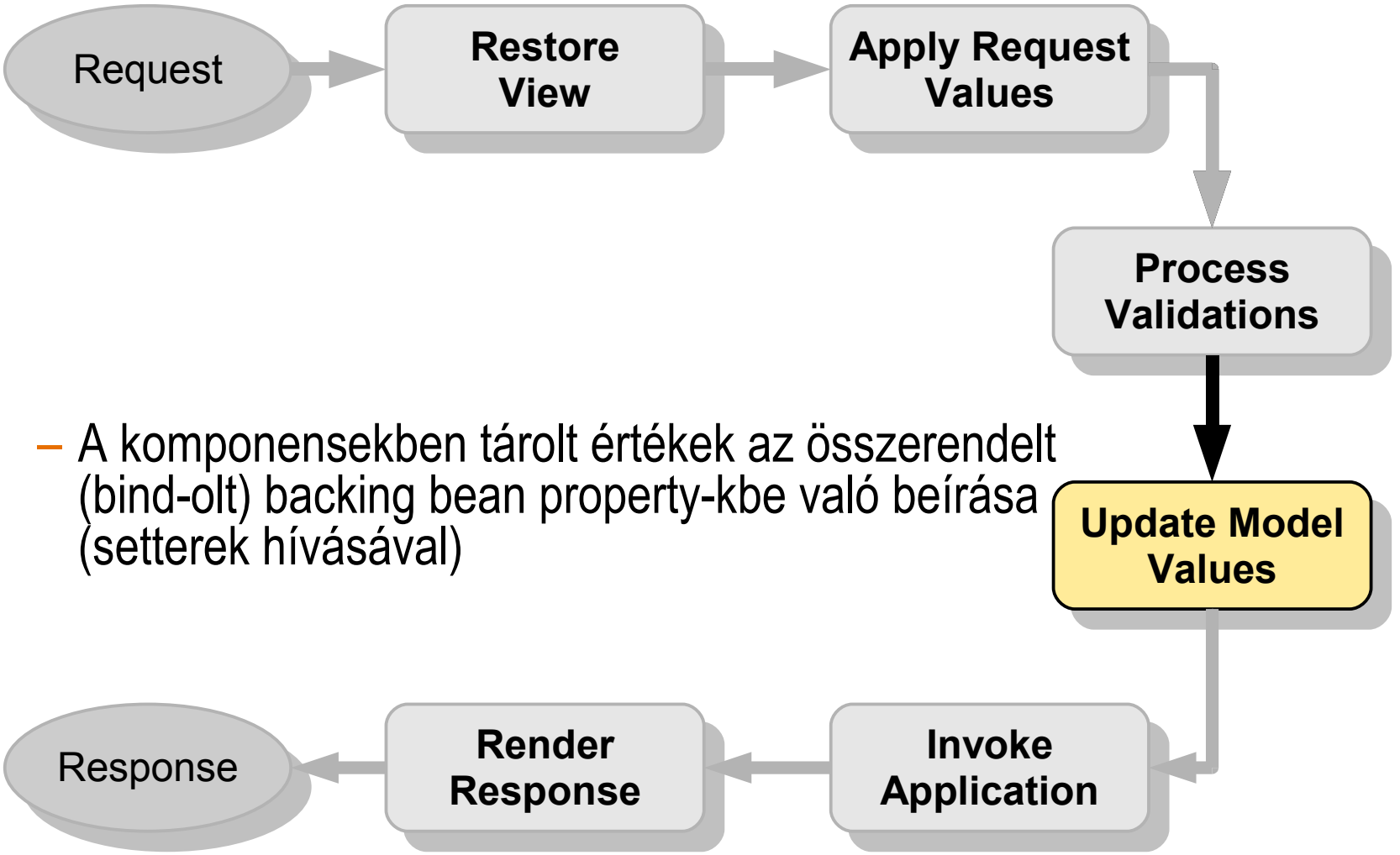
- A komponensek eseményeket generálnak, a GUI-ban megszokott módon kezeljük az eseményeket (observer pattern, FacesListener, FacesEvent)
- Minden fázisban keletkezhetnek események, de általában másik fázisban kell őket kezelni: a gomb megnyomása esetén az ActionEvent már az *Apply Request Values* fázisban létrejön, de csak az *Invoke Application* fázisban kezeljük
- ActionEvent – alapértelmezett listener

A JSF kérésfeldolgozás – visszaküldés



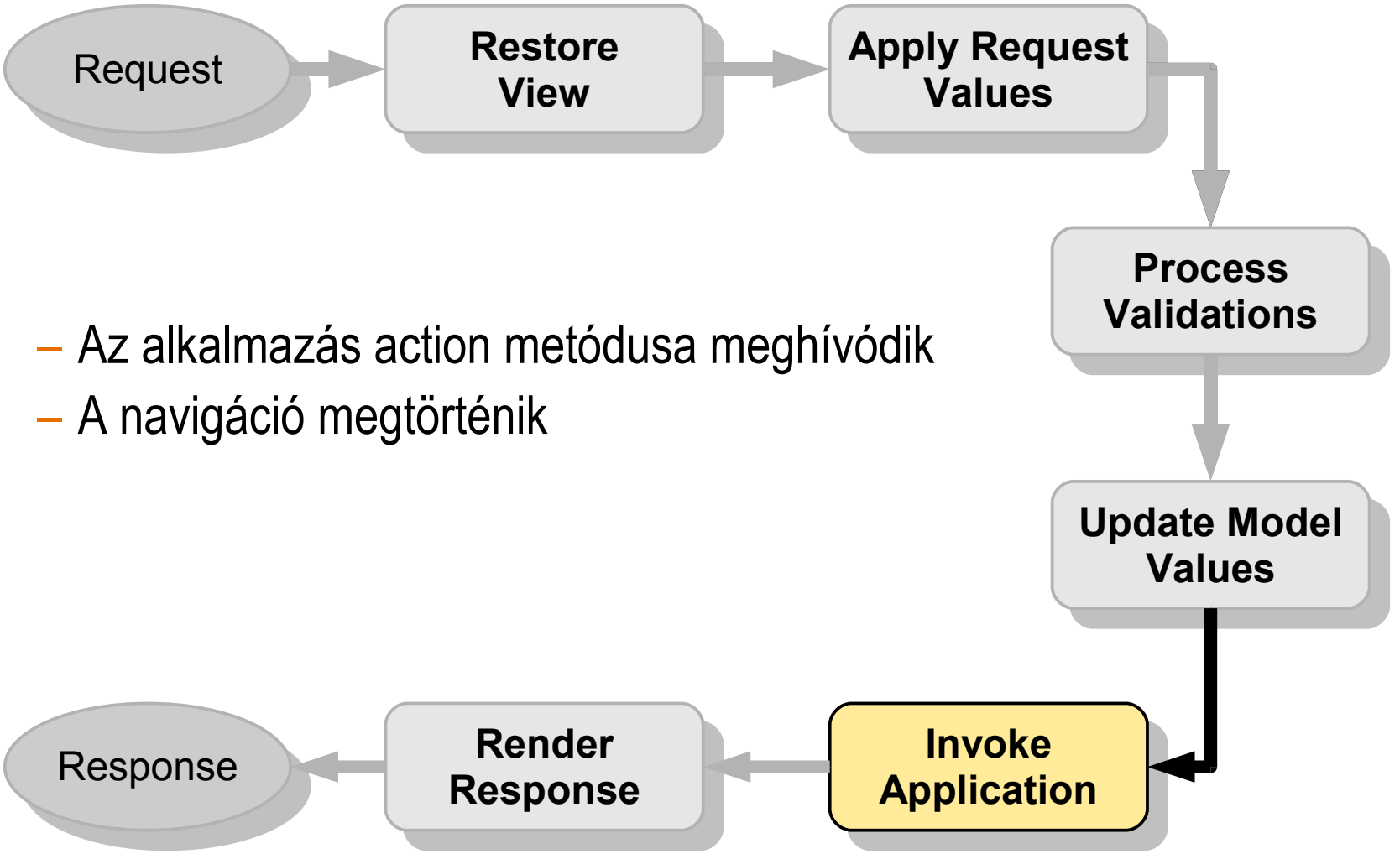
- Konvertált értékek validálása

A JSF kérésfeldolgozás – visszaküldés



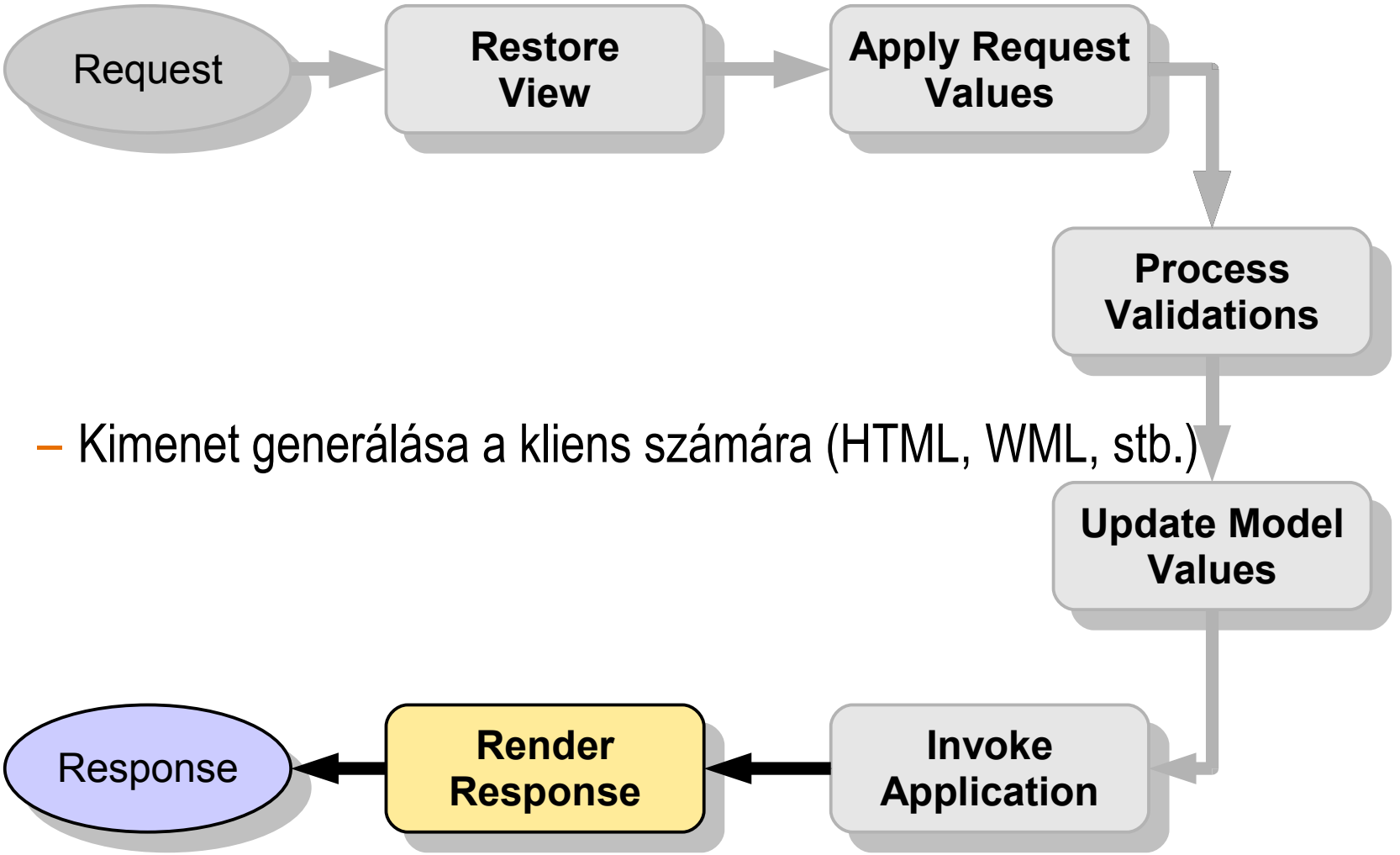
- A komponensekben tárolt értékek az összerendelt (bind-olt) backing bean property-kbe való beírása (setterek hívásával)

A JSF kérésfeldolgozás – visszaküldés



- Az alkalmazás action metódusa meghívódik
- A navigáció megtörténik

A JSF kérésfeldolgozás – visszaküldés

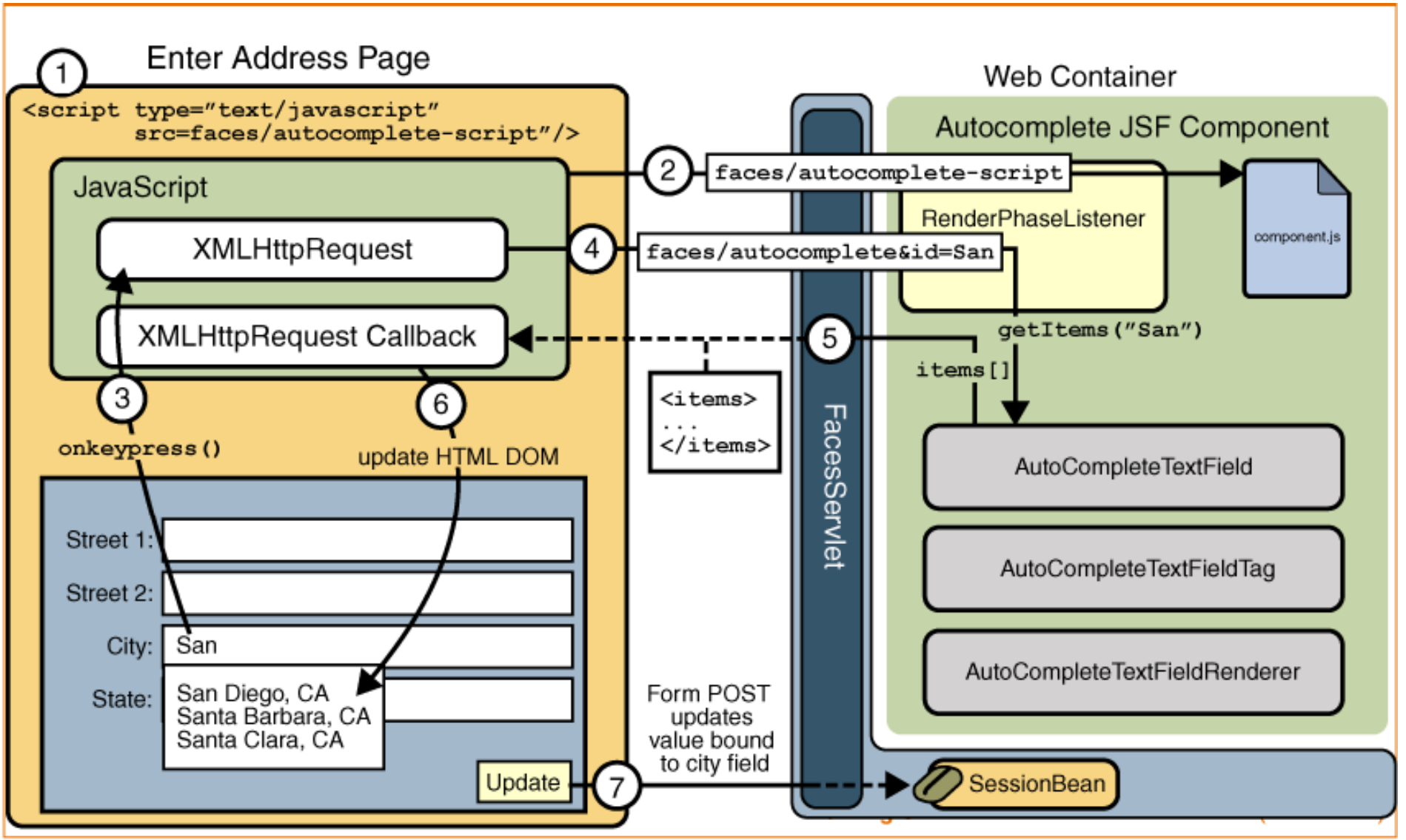


- Kimenet generálása a kliens számára (HTML, WML, stb.)

A JSF és az AJAX

- A JSF AJAX barát, mert jellemzője:
  - A rugalmas és bővíthető komponens modell,
  - A jól definiált kérésfeldolgozási ciklus
  - A rugalmas és bővíthető renderelési modell
- Alapelvek, melyek lehetővé teszik az AJAX használatát
  - Egységbe zárás: el lehet rejteni a JavaScript-et az oldal készítője elől, de meg lehet mutatni a komponensfejlesztőnek
  - Állapot menedzsment: könnyű szinkronban tartani a szerver oldali és a kliens oldali állapotot
- JSF és AJAX integrációja – JSF 2.0

A JSF és az AJAX



A JSF barátsága a vizuális fejlesztőeszközökkel

- Komponens alapú
  - A rugalmas és bővíthető komponens modell,
  - A komponensek JavaBeanek (design time, runtime választás)
- A response gyártás a specifikáció szerint fel van készítve a fejlesztőeszközökre
  - A ResponseWriter osztály olyan, hogy a kimenet készítése közben mindig meg kell mondani, hogy melyik komponenshez tartozik az adott generált kód (HTML, WML, XML, stb.)

A NetBeans IDE 5.5 JSF támogatása

- Leíró XML-ek ismerete, kódkiegészítés, validáció
  - web.xml
  - faces-config.xml
- Komponens Tag library-k használatában való segítség, deployolás nélküli validáció
- Managed Bean-ek nyilvántartása - EL kifejezések kiegészítése

A NetBeans Visual Web Pack plugin

- Eredetileg Java Creator Studio néven futó termékből NetBeans plugin lett – J2EE 5.0, JSF 1.2 támogatással
- Vizuális oldal szerkesztés – rapid fejlesztés
- Komponens hierarchia vizuális megjelenítése
- Komponensek tulajdonságainak szerkesztése testreszabott property szerkesztőkkel
- Saját (szabványban leírt komponensektől jóval kifinomultabb) komponens könyvtár
- Bővíthető komponenskészlet

Összegzés

- Gyors és hatékony fejlesztési mechanizmus
- Szabványos, komponens alapú technológia – minden alkalmazásszerver része J2EE 5.0 óta
- Rapid fejlesztőeszközbarát – és már létezik ilyen: Netbeans Visual Web Pack plugin
- Extrém rugalmas: tetszőleges része testreszabható
- JSF + AJAX: egymásnak teremtettek
- Kérdés: Milyen meredek a JSF tanulási görbéje?

Kérdések/Demo