

Java™ Portlets with Liferay® and JSF

Trayan Iliev

IPT – Intellectual Products & Technologies
e-mail: tiliev@iproduct.org
web: <http://www.iproduct.org>

Liferay® is a registered trademark of Liferay, Inc. Oracle®, Java™ and EJB™ are trademarks or registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners. Liferay® е регистрирана търговска марка на Liferay, Inc. Oracle®, Java™ и EJB™ са търговски марки на Oracle и/или негови подразделения. Всички други търговски марки са собственост на техните притежатели.



Licensed under the **Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License**

Slide 1

Въпрос 1



Каква е разликата между уеб сайт
и бизнес портал?



Предимства на порталите (1)

- Уеб порталите осигуряват значителни предимства за бизнеса (<http://portals.apache.org/>):
 - осигуряват **входна точка** за достъп за всички служители, партньори и клиенти
 - предлагат достъп до бизнес функционалността прозрачно и **независимо от устройството и местоположението**
 - порталите са **гъвкави** – те могат да съществуват под формата на B2E intra-nets, B2B extra-nets или B2C inter-nets



Предимства на порталите (2)

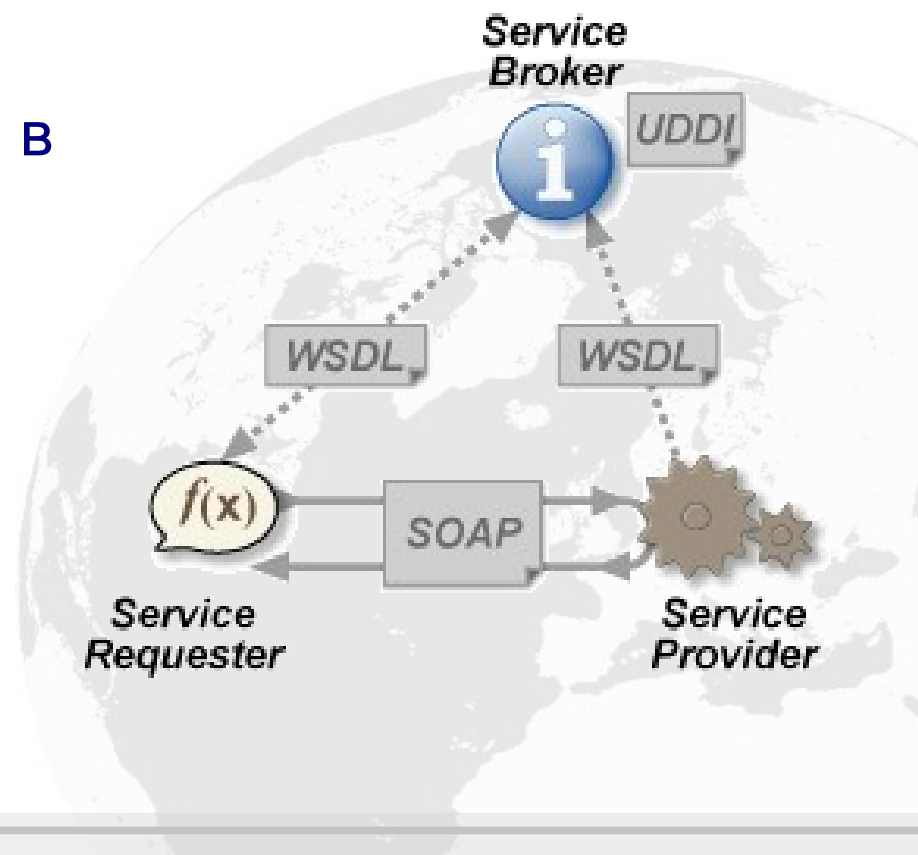
- Уеб порталите осигуряват значителни предимства за бизнеса (<http://portals.apache.org/>):
 - могат да бъдат комбинирани в портални мрежи, които обхващат цялата бизнес екосистема на организацията
 - понеже осигуряват потребителски интерфейс („front end“) за различни уеб услуги, те позволяват лесно интегриране на хетерогенни съществуващи приложения и са отворени към бъдещето (future-proof)
 - част от по-широката визия за реализация на Service Oriented Architecture (SOA) в бизнес организациите



SOA == Уеб услуги ?

Уеб услугите са:

- компоненти за изграждане разпределени приложения в SOA архитектурен стил
- комуникират използвайки отворени протоколи
- само-описателни и само-съдържащи се
- могат да бъдат откривани използвайки UDDI или ebXML регистри



Архитектура ориентирана към услуги (SOA) – дефиниции

- **OASIS** Reference Model for Service Oriented Architecture 1.0
Service Oriented Architecture (SOA) is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations.
- **Wikipedia:** Service-oriented architecture (SOA) is a flexible set of design principles used during the phases of systems development and integration in computing. A system based on a SOA will package functionality as a suite of interoperable services that can be used within multiple, separate systems from several business domains.



Архитектура ориентирана към услуги (SOA) – дефиниции (2)

Webopedia: Service-oriented architecture (SOA) – an application architecture in which all functions, or services, are **defined using a description language** and have **invokable interfaces** that are called to perform **business processes**. Each interaction is **independent** of each and every other interaction and the interconnect protocols of the communicating devices (i.e., the infrastructure components that determine the communication system do not affect the interfaces). Because **interfaces are platform-independent**, a client from any device using any operating system in any language can use the service.

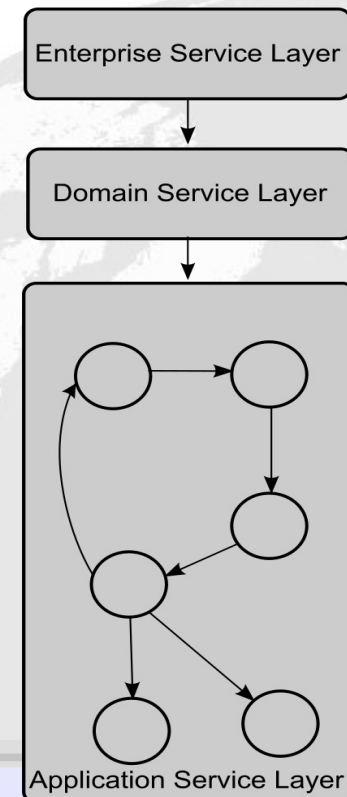
Though built on similar principles, **SOA is not the same as Web services**, which indicates a collection of technologies, such as SOAP and XML. **SOA is more than a set of technologies** and runs independent of any specific technologies.



Архитектура ориентирана към услуги (SOA) – ОСНОВНИ ПРИНЦИПИ

Съгласно Yvonne Balzer - IBM (<http://www.ibm.com/developerworks/webservices/library/ws-improvesoa/>):

- Многократно използване, гранулярност, модулност, композируемост, компонентна ориентация, съвместно опериране (interoperability);
- Съответствие със стандартите (както общи, така и специфични за съответната индустрия);
- Възможност за идентификация и категоризация, осигуряване и доставка, наблюдение и проследимост на услугите.



Източник: <http://en.wikipedia.org/wiki/File:Soa-layers.svg>,
автор: Feravoon, лиценз: Public Domain

Архитектура ориентирана към услуги (SOA) – основни принципи (2)

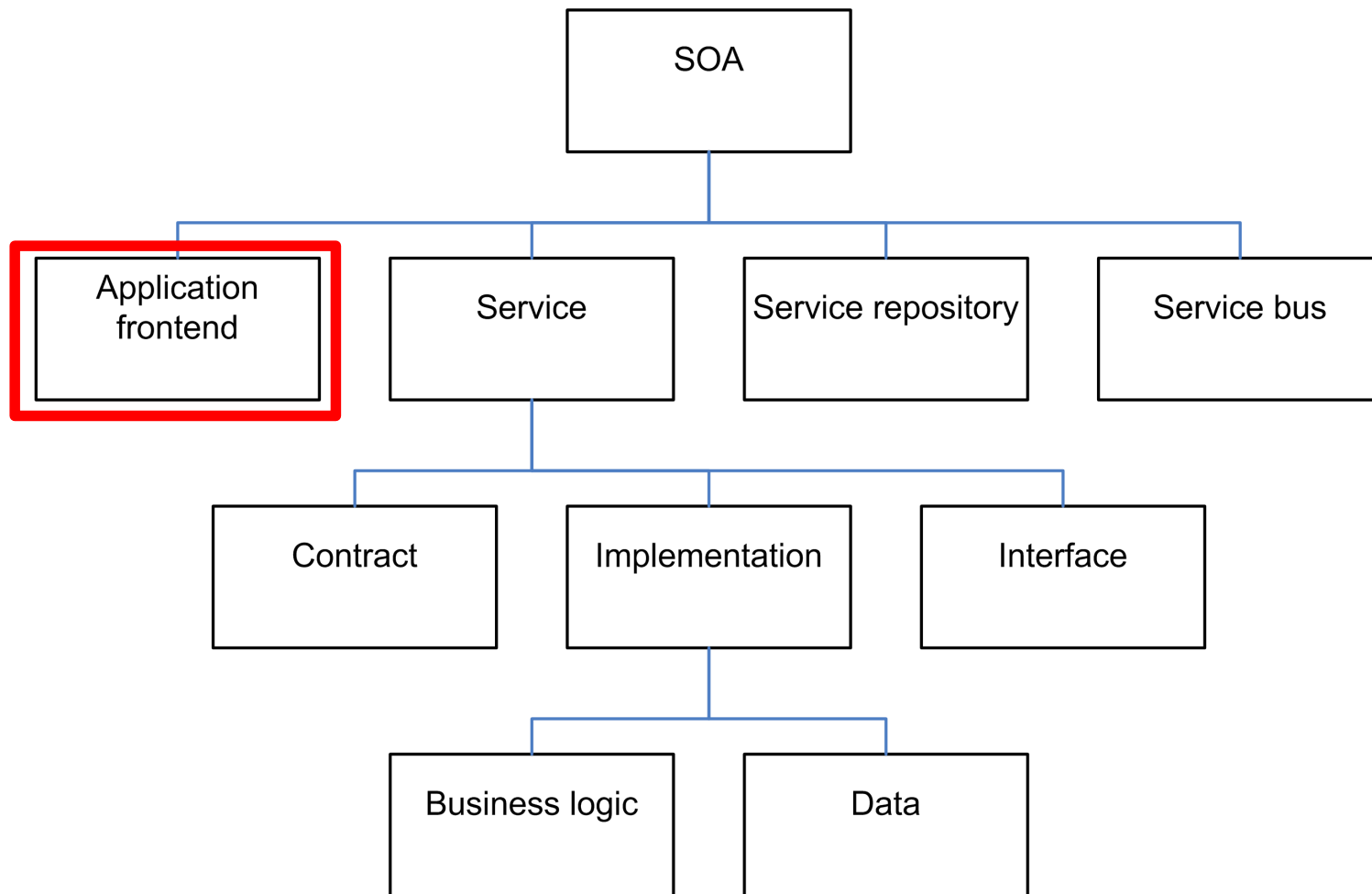
Съгласно <http://www.soaprinciples.com/>:

- Standardized Service Contract
- Service Discoverability
- Service Loose Coupling
- Service Composability
- Service Abstraction
- И още:
- Service Reusability
- Service Optimization
- Service Autonomy
- Service Relevance
- Service Granularity
- Service Encapsulation
- Service Statelessness
- Service Location

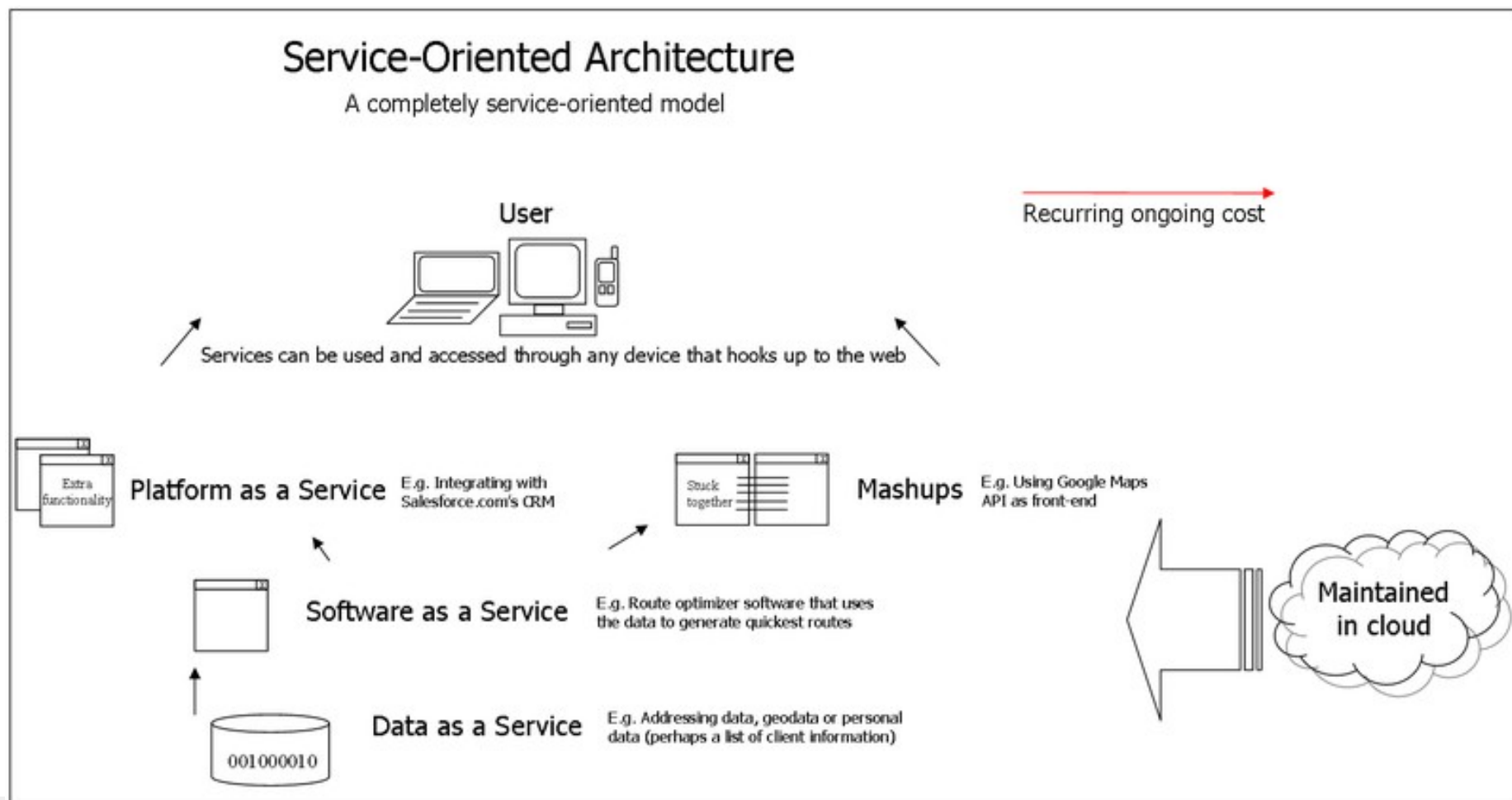
Transparency

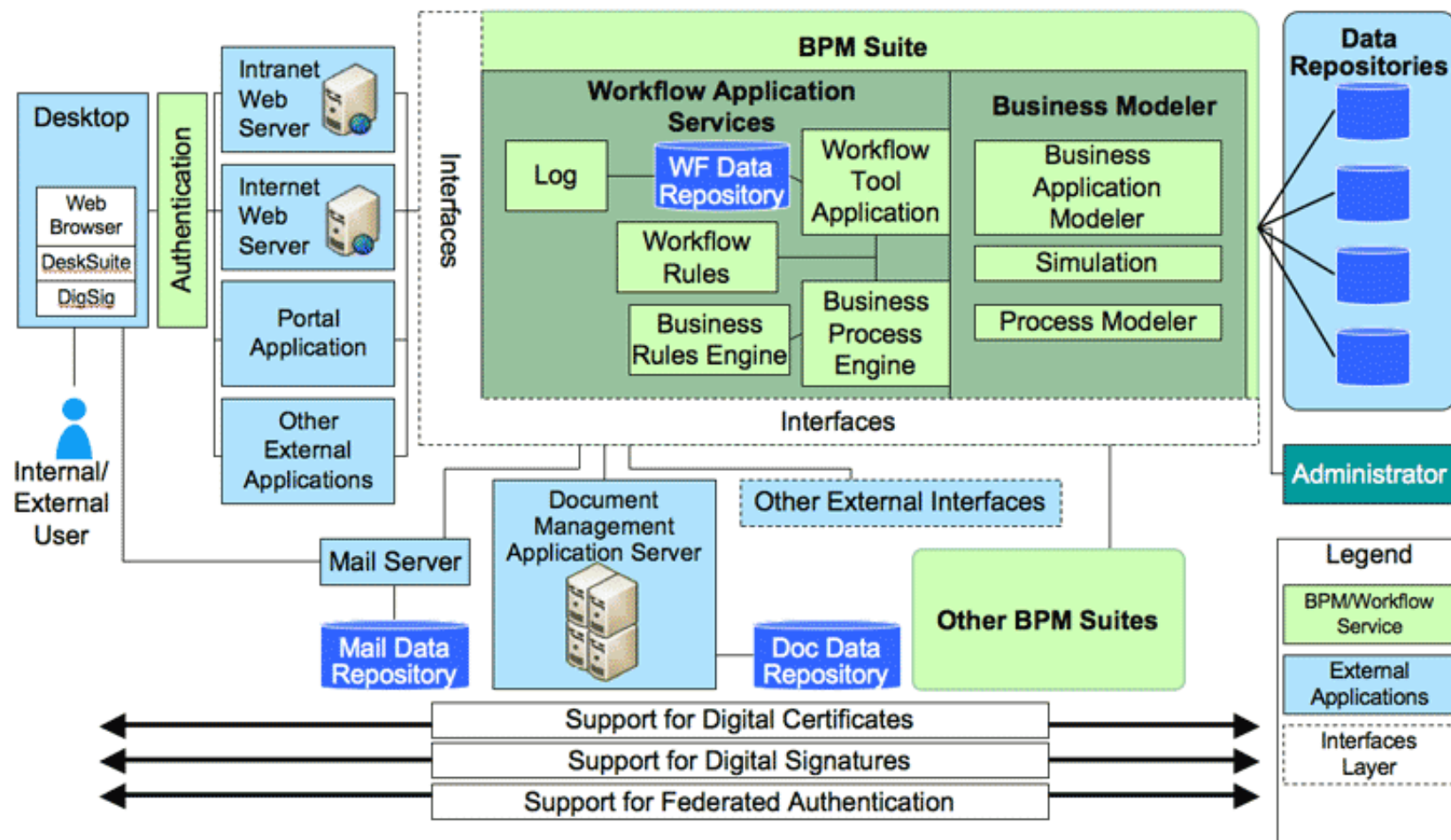


SOA – ОСНОВНИ ПОНЯТИЯ



Service Oriented Architecture (SOA)





Пример за бизнес портал, който демонстрира как инструментите за управление на бизнес процеси (BPM) могат да бъдат използвани за реализация на бизнес процеси чрез оркестрация и хореография на дейности изпълнявани от хора и софтуерни системи (Източник: National Institute of Health (2007). Business Process Management (BPM) Service Pattern – <http://enterprisearchitecture.nih.gov/ArchLib/AT/TA/WorkflowServicePattern.htm>)

Реализация на SOA - Java SE/EE 6 APIs

Съгласно Java EE спецификацията:

- Web Services (SOAP)
 - ~~Java API for XML-based RPC (JAX-RPC)~~
 - Java API for XML Web Services (JAX-WS)
 - Java Architecture for XML Binding (JAXB)
 - SOAP with Attachments API for Java (SAAJ)
 - Java API for XML Registries (JAXR)
- RESTful Web Services
 - Java API for RESTful Web Services - JAX-RS



Въпрос 2



Какво липсва в разгледаните до момента технологии и стандарти за реализация на SOA?



Отговор:



THE User

- Само уеб услугите не са достатъчни ...
- Необходимо е потребителят да може да ги ползва и персонализира
- Необходимо е да може да ги комбинира **динамично** според уникалните си нужди и задачи

Решението: Java™ портлети!

- През 2002 Java™ общността стартира пионерски усилия за стандартизация на **уеб базирани презентационно-ориентирани компоненти (услуги) == портлети**
- Портлетите могат лесно да бъдат комбинирани в **портлетни приложения (Portlet Applications)** за изграждане на бизнес портали на базата на общ стандарт, който дефинира тяхното взаимодействие с портала и помежду им – **JSR 168: Java Portlet Specification**
- През 2005 **JSR 168** е наследен от подобрена спецификация – **JSR 286: Portlet Specification 2.0** (ползваща се от въведените в Java 5 езикови улеснения като например използването на **анотации**)





IPT - Intellectual Products & Technologies

Knowledge makes you free ∞

Knowledge makes you free ∞

Welcome!

VOD

SemanticContent

UGC

Welcome

Temp

Admin

Semantic Demo

MyChat2

JSF 2.0 Jobs

JFS 2.0 Events

JSF 2.0 Ajax Push

VOD Chooser

A-Z

New

Top 10

Most Seen

Search

Cart

FAQ

Help

Rubrics

IPT Ads and Intros

Black Sea Moments

Black Sea Panorama

Black Sea Views

Veleka River

IPT Announcements

Videos

A Boulder in the Water

There is a Boulder in the Sea near
[read more ...](#)

Sunset 1

The Sunset on the Butamiata Beach
[read more ...](#)

Wild Sea 1

Wild Black Sea near Veleka Beach
[read more ...](#)

IPT VOD Player

IPT Video On Demand



Add to Cart

Visited 1 times

★☆☆☆☆

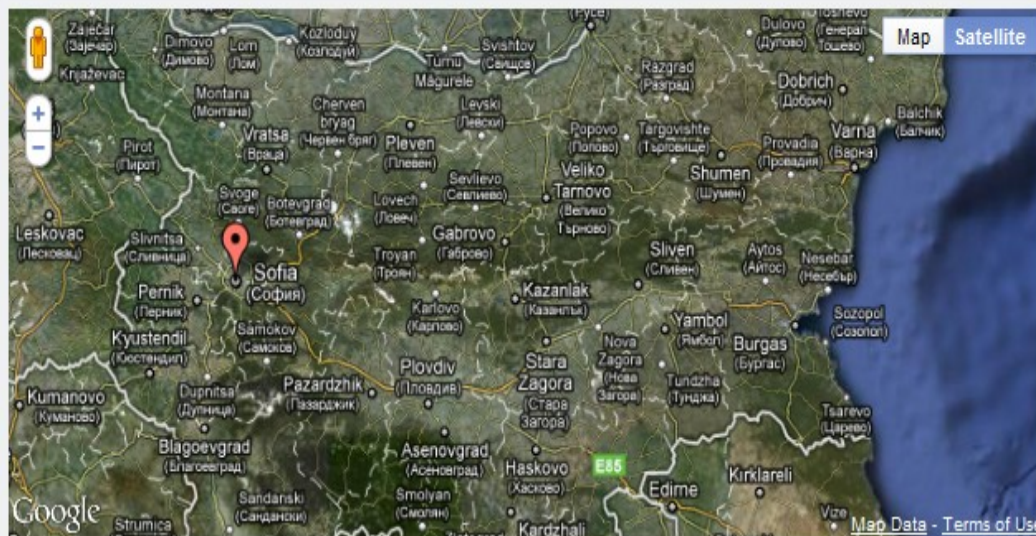
Video Metadata

[Sign In](#)

Forum

JS Mashup Portlets

My Map Portlet with Events



Sofia

[Show Location](#)

Shared Render Parameter - location: Sofia

My Weather Portlet with Events

Weather for Sofia, Bulgaria

Tue, 20 Sep 2011 2:29 pm EEST

Current Conditions:

Mostly Cloudy, 70F

21° C



Forecast:

20 Sep 2011-Heavy Rain .High:24° C Low:15° C

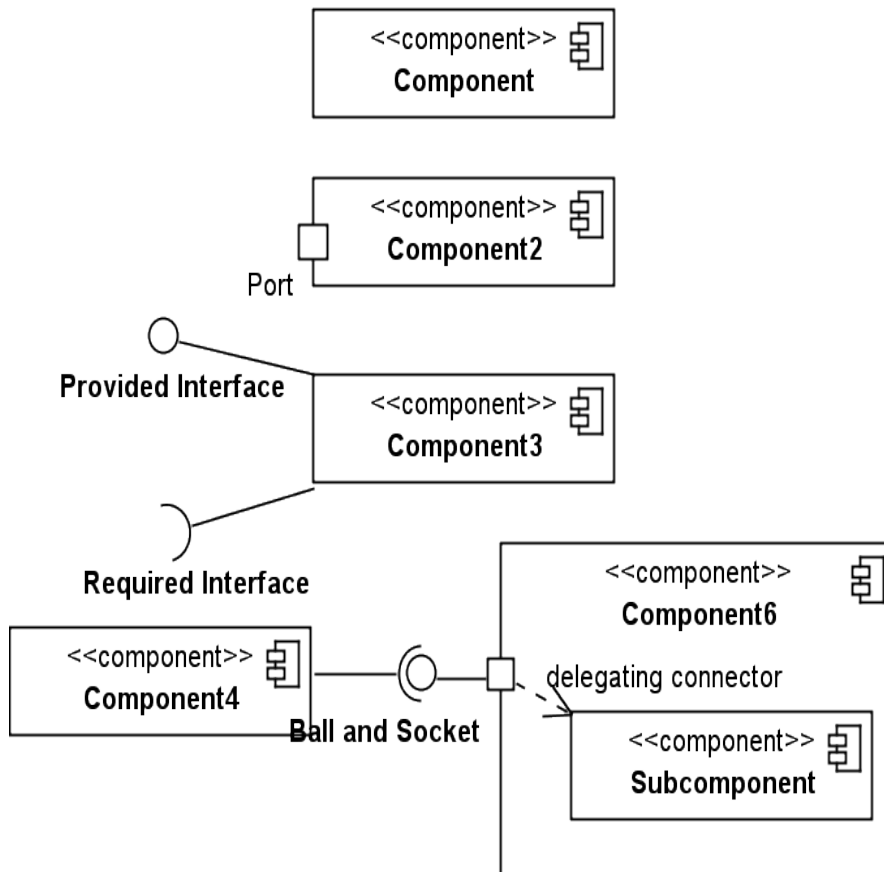
21 Sep 2011- Heavy Rain .High:23° C Low:13° C

Enter City Name

[Show Weather](#)

Shared Render Parameter - location: Sofia

Въпрос 3



Какво всъщност
означава „компонент“
в софтуерното
инженерство?

Отговор:

“Компонентите не са технология. Хората, които разработват технологии трудно разбират това. Компонентите се отнасят до това как клиентите искат да се отнасят към софтуера. Те искат да могат да го купуват на части и да могат да го ъпгрейдват както стерео уредбата си например. Те искат новите части да работят безпроблемно със старите и да могат да правят ъпгрейд по свой собствен график. Те искат да могат да смесват компоненти от различни доставчици. Това е едно много смислено изискване. Просто е трудно да бъде реализирано.”

Ралф Джонсън, <http://www.c2.com/cgi/wiki?DoComponentsExist>



Еволюцията на бизнес порталите (1)

- Портлетите стават все по-популярни при разработката на портали, защото те позволяват лесно споделяне, комбиниране и интеграция на уеб приложения и компоненти от различни доставчици в един персонализиран бизнес портал
- Възниква нов "портлетно-базиран" стил при разработката на уеб приложения. Портлетните приложения (Portlet Applications - PA) са по-разпределени, гъвкави, лесни за вграждане и повторно използване в сравнение с класическите „монолитни“ уеб приложения, които познаваме от години.



Еволюцията на бизнес порталите (2)

- **РА** могат да използват асинхронни заявки за данни към сървъра и могат динамично да осъвременяват визуализираната информация в отговор (а понякога и предвиждайки) нуждите на потребителя.
- Съгласно актуалната **Portlet Specification 2.0**:
 - **РА** типично се състоят от множество различни портлети
 - Различните портлети могат да комуникират използвайки **споделени параметри (shared parameters)** или **публикуване/абониране за събития (publish/subscribe events)** и могат да правят динамични заявки за ресурси към сървъра (**AJAX**)



Какво е уеб портал?

Съгласно Portlet Specification 2.0 (<http://www.jcp.org/en/jsr/detail?id=286>):

- Порталите са уеб приложения, които хостват презентационния слой на информационните системи, позволяват агрегация на съдържание, произведено от различни уеб базирани презентационни компоненти (портлети) от различни източници, осигуряват персонализация и управление на това съдържание, както и множество от услуги (управление на потребителите, аутентификация, авторизация, управление на правата и др.)



Какво е портлет? (1)

Съгласно **Portlet Specification 2.0** (<http://www.jcp.org/en/jsr/detail?id=286>):

- Портлетът е **уеб-базиран компонент на приложението**, който осигурява специфична част от съдържанието (**информация или услуга**), която може да бъде включена като част от **портална страница**.
- **Контейнерът за портлети (Portlet Container - PC)** – управлява различните портлети, маршрутизира заявките за страници към тях и връща техните отговори към портала, за да бъде динамично конструиран отговорът на заявка за порталната страница.
- Портлетите са **“pluggable”** компоненти в **презентационния слой** на приложението



Какво е портлет? (2)

Съгласно **Portlet Specification 2.0** (<http://www.jcp.org/en/jsr/detail?id=286>):

- Портлетите генерират **page markup** фрагменти с използване на :
 - HTML
 - XHTML
 - WML
 - други маркъп езици
- Отговорът на портлета (**Portlet Response**) може да бъде персонализиран за конкретен потребител използвайки портлетната конфигурация (**Portlet Configuration**) за този потребител



What Is Portlet Container?

Съгласно Portlet Specification 2.0 (<http://www.jcp.org/en/jsr/detail?id=286>):

- Portlet Container (PC) управлява manages жизнения цикъл на различните портлети, предоставяйки им обща среда за изпълнение (runtime environment), и дълготрайно съхраняване на информацията (persistent storage) за портлетните предпочитания (Portlet Preferences)
- PC не агрегира резултатното съдържание – отговорност на портала е да направи това
- PC и порталът могат да бъдат изградени като един общ сървър или да бъдат осигурени като различни компоненти



Разлики между портлети и сървлети (1)

Съгласно **Portlet Specification 2.0** (<http://www.jcp.org/en/jsr/detail?id=286>):

- Портлетите генерират само **маркъп фрагменти** (а не цели веб страници), които да бъдат агрегирани от портала
- Портлетите могат да обработват различни типове заявки:
 - **Action Request** – променя състоянието на портлета
 - **Event Request** – променя състоянието на текущия или на някои от останалите портлети при генериране на събития в рамките на Action Request -а
 - **Render Request** – връща маркъп в зависимост от текущото състояние на портлета
 - **Resource Request** – връща ресурс (асинхронно)



Разлики между портлети и сървлети (2)

Съгласно **Portlet Specification 2.0** (<http://www.jcp.org/en/jsr/detail?id=286>):

- Клиентите не взаимодействат с портлетите директно, а чрез портала/портлетния контейнер (PC)
- Навигацията се осъществява с използване на URL-и специфично конструирани за целта използвайки Portlet API (**action URLs, render URLs, resource URLs**)
- Портлетите имат различни **Portlet Modes (PM)** – View /Edit /Help, и др., както и **Window States (WS)**
- Портлетите могат да съхраняват и достъпват **persistent configuration** данни



Разлики между портлети и сървлети (3)

Съгласно Portlet Specification 2.0 (<http://www.jcp.org/en/jsr/detail?id=286>):

- Инстанциите на портлет (Portlet Instances - PI) могат да бъдат включвани повече от веднъж в една и съща страница (евентуално с различни конфигурации)
- Портлетите могат да достъпват информация за потребителския профил (User Profile information)
- Портлетите могат да изпращат и получават събития (Events) генерирани от други портлети или от контейнера
- Портлетите могат да съхраняват данни за сесията в два различни обхвата – application-wide или portlet-private сесийни атрибути



Връзки между портлети и сървлети (1)

Съгласно **Portlet Specification 2.0** (<http://www.jcp.org/en/jsr/detail?id=286>):

- **ServletContext --> PortletContext**
 - javax.servlet.ServletContextListener
 - javax.servlet.ServletContextAttributeListener
- **HttpSession --> PortletSession**
 - javax.servlet.http.HttpSessionActivationListener
 - javax.servlet.http.HttpSessionAttributeListener
 - javax.servlet.http.HttpSessionBindingListener

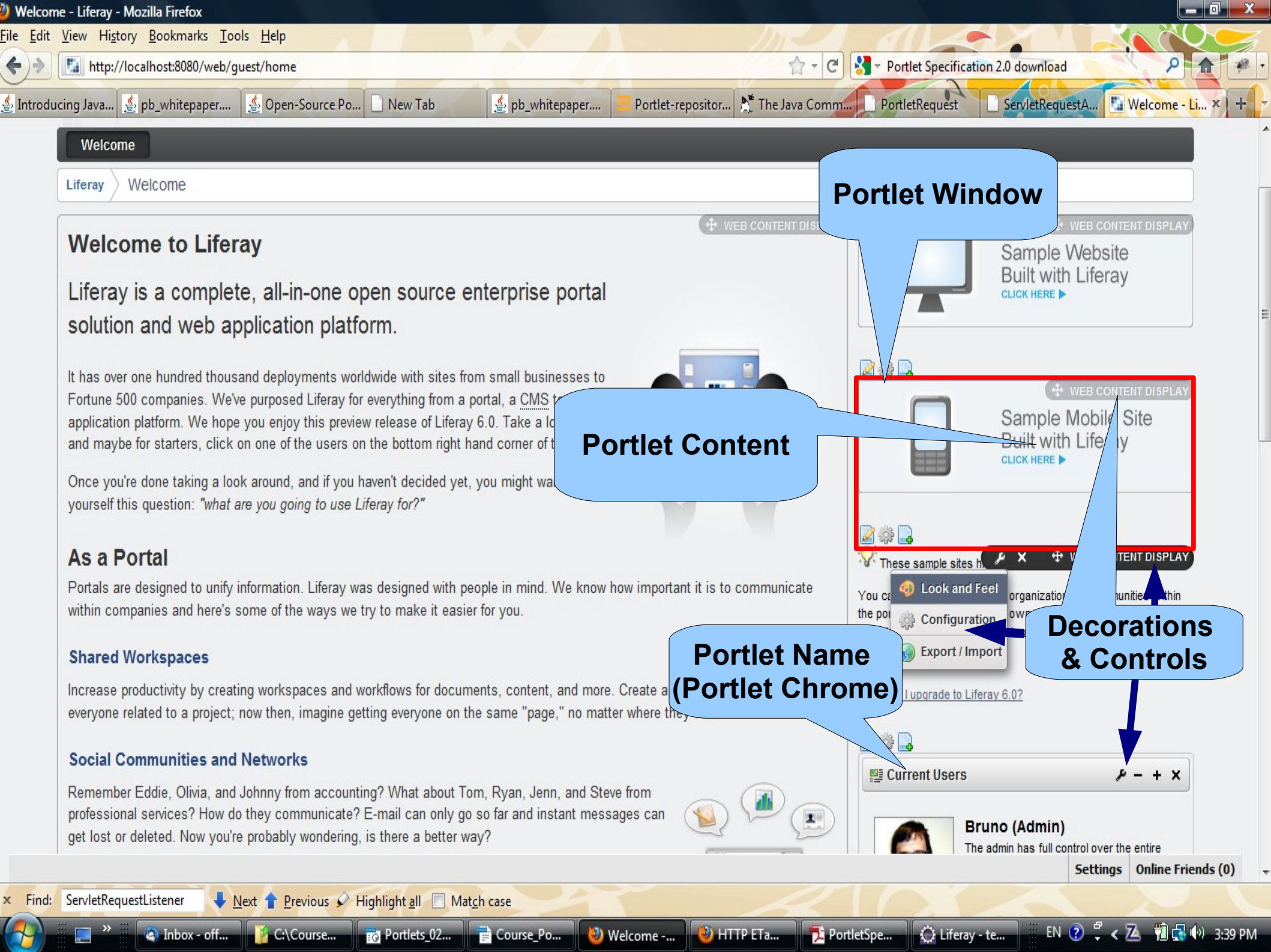


Връзки между портлети и сървлети (2)

Съгласно **Portlet Specification 2.0** (<http://www.jcp.org/en/jsr/detail?id=286>):

- **HttpServletRequest** mirrors **PortletRequest**
 - `javax.servlet.ServletRequestListener`
 - `javax.servlet.ServletRequestAttributeListener`
- **PortletRequest** не е споделен а е само “отрзяван огледално” от съответен **ServletRequest** създаден автоматично от контейнера





Portlet Window

Portlet Content

Portlet Name
(Portlet Chrome)

Decorations
& Controls

Жизнен цикъл на портлетните приложения (1)

- **Portlet Container (PC)** зарежда и инстанцира индивидуалните портлети:
 - когато се стартира **Portlet Application (PA)**
 - “мързеливо” (lazy), когато бъде приета първата заявка към портлета
- **PC** инициализира портлетните инстанции преди извикване:
 - викайки **init(PortletConfig config)** метод на **Portlet** интерфейса, предавайки инициализиращи параметри, локализиран **ResourceBundle** и др.
 - Може да хвърли **PortletException** или **UnavailableException** - портлетът се унищожава **без** да се вика метода **destroy()**



Жизнен цикъл на портлетните приложения (2)

- Когато **Portlet Container (PC)** определи, че инстанцията на портлета вече не е необходима, той извиква метода **destroy()** на **Portlet** интерфейса, след като всички активни заявки бъдат обработени
 - Ако се генерира **RuntimeException**, то PC счита инстанцията на портлета за унищожена успешно
 - не бива да се ползват **finalizers**
- По време на активната работа портлетната инстанция се използва многократно (Flyweight design pattern) за обслужване на множество заявки от различни клиенти, дааните за заявката и клиента се осигуряват **“on the fly”**



Основни понятия свързани с портлетите

- **Portlet Definition** – състои се от **preference attributes** и техните стойности по подразбиране (специфицирани в портлетния **deployment descriptor – portlet.xml**), използва се за създаване на един или повече **PortletPreferences** обекти (връщани от **PortletRequest.getPreferences()**)
- **Portlet Entity** – абстрактно понятие = асоциация на един или повече **PortletPreferences** обекти с портлета
- **PortletPreferences** могат да бъдат добавяни, изтривани или модифицирани
- **Portlet Window** има **window state**, **portlet mode**, **render parameters**, **portlet-scoped session state** + **unique ID** (достъпно чрез метод **PortletRequest.getWindowID()**)



JSR 168: Java™ Portlet API

- **init()** - извикван след като портлетът бъде инстанциран
- **destroy()** - извикван преди контейнерът да унищожи портлета
- **processAction()** - извикван когато потребителят промени състоянието на портлета
- **render()** - извикван при всяко рендериране на портлета / портлетния прозорец
- **doView()** - извикван от **render()** когато сме във View режим
- **doEdit()** - извикван от **render()** когато сме в Edit режим
- **doHelp()** - извикван от **render()** когато сме в Help режим
- Състояния на портлетния прозорец: **MINIMIZED, NORMAL, MAXIMIZED**



Нови възможности в JSR 286: Java™ Portlet 2.0 API

- **Events** – дават възможност за слабо-свързана (loosely-coupled) комуникация между различни портлети
- **Shared render parameters** – осигуряват синхронизация на визуализациите (views) представяни от различни портлети
- **Resource serving requests** – посвоява асинхронни заявки за ресурси в рамките на портлета
- **Portlet filters** – аналог на Servlet filters, позволява обработка или трансформация на портлетни заявки и отговори “on-the-fly”
- **AJAX support** – използвайки ресурсни заявки генерирани от JS асинхронно, без презареждане на страницата



JSR 286: Java™ Portlet 2.0 API: нови анотации и метода на класа GenericPortlet

- Нови анотации:
 - `@ProcessAction` – обозначава метод обработващ действие на потребителя (action)
 - `@ProcessEvent` – обозначава метод обработващ събития
 - `@RenderMode` – обозначава метод връщащ `RenderResponse` за конкретен `PortletMode`
- Нови методи:
 - `processEvent(EventRequest req, EventResponse resp)` - обработва събитие генерирано от този или друг портлет
 - `serveResource(ResourceRequest rr, ResourceResponse rsp)`



Sample Portal Page with Portlets: A, B, C and D

Portlet A

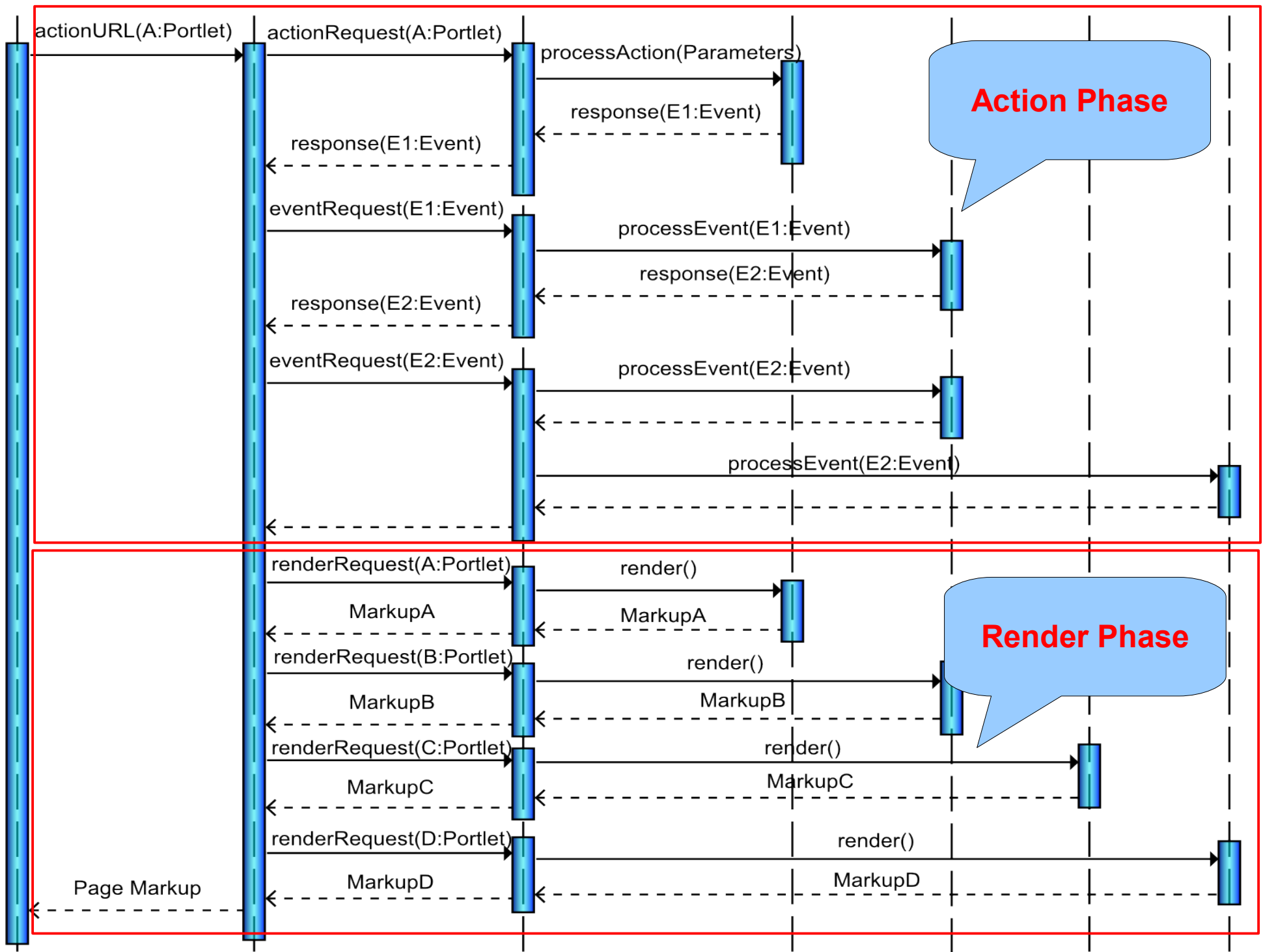
Portlet B

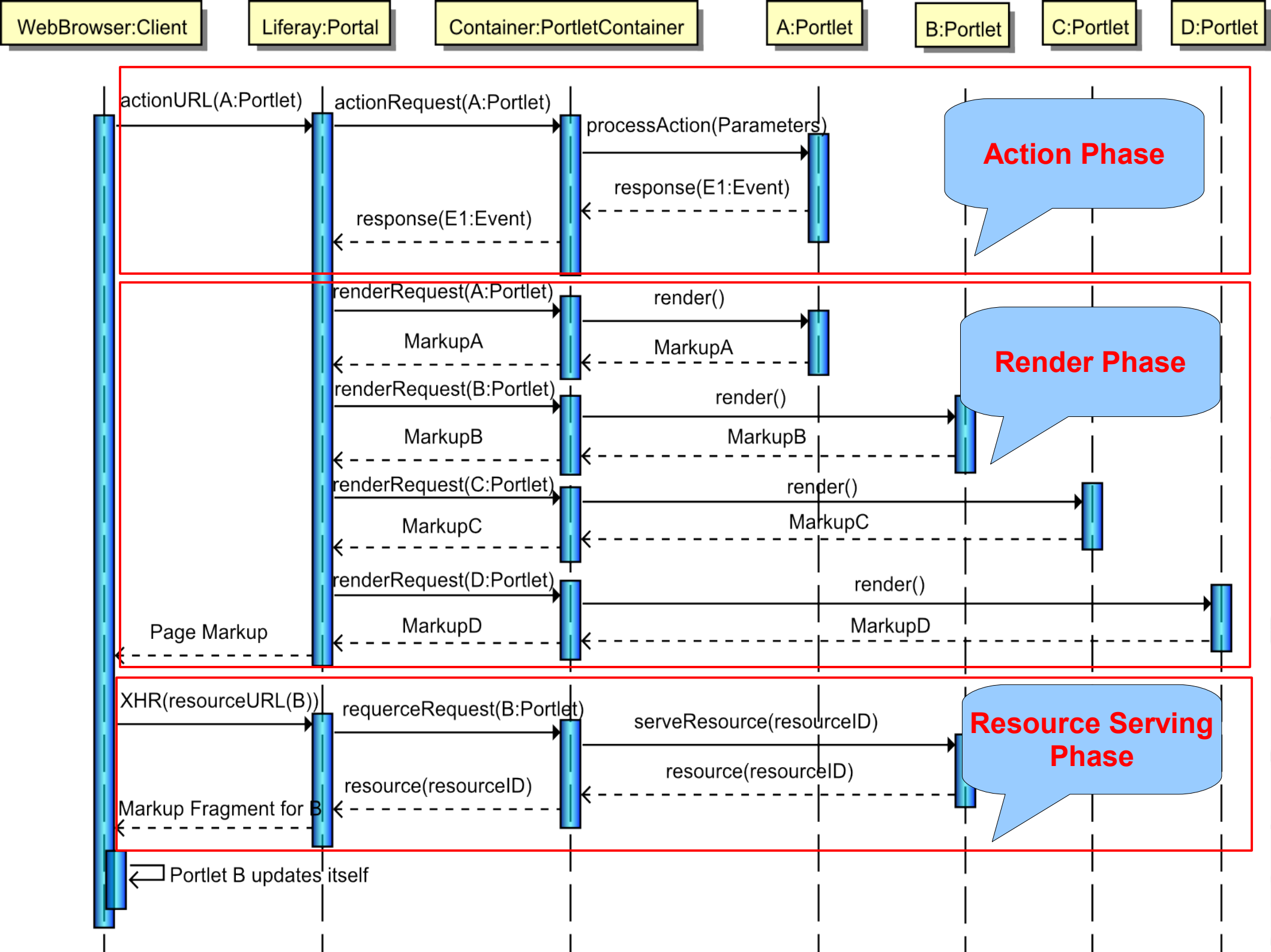
Portlet C

Portlet D



Licensed under the Creative Commons Attribution-NonCommercial-NoDerivs 3.0
Unported License





Portlet 2.0 Deployment Descriptor - portlet.xml(1)

```
<?xml version="1.0"?>
<portlet-app
  version="2.0"
  xmlns="http://java.sun.com/xml/ns/portlet/portlet-app_2_0.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/portlet/portlet-
app_2_0.xsd http://java.sun.com/xml/ns/portlet/portlet-app_2_0.xsd"
>
  <portlet>
    <portlet-name>test-portlet</portlet-name>
    <display-name>Test Portlet</display-name>
    <portlet-class>com.test.NewPortlet</portlet-class>
```



Portlet 2.0 Deployment Descriptor - portlet.xml(2)

```
<init-param>
  <name>view-jsp</name>
  <value>/html/newportlet/view.jsp</value>
</init-param>
<init-param>
  <name>edit-jsp</name>
  <value>/html/newportlet/edit.jsp</value>
</init-param>
<init-param>
  <name>help-jsp</name>
  <value>/html/newportlet/help.jsp</value>
</init-param>
```



Portlet 2.0 Deployment Descriptor - portlet.xml(3)

```
<expiration-cache>0</expiration-cache>
<supports>
  <mime-type>text/html</mime-type>
  <portlet-mode>VIEW</portlet-mode>
  <portlet-mode>EDIT</portlet-mode>
  <portlet-mode>HELP</portlet-mode>
</supports>
<resource-bundle>content/Language</resource-bundle>
<portlet-info>
  <title>Test Portlet</title>
  <short-title>NewPortlet</short-title>
  <keywords></keywords>
</portlet-info>
```



Portlet 2.0 Deployment Descriptor - portlet.xml(4)

```
<security-role-ref><role-name>administrator</role-name>
</security-role-ref>
<security-role-ref><role-name>guest</role-name>
</security-role-ref>
<security-role-ref><role-name>power-user</role-name>
</security-role-ref>
<security-role-ref><role-name>user</role-name>
</security-role-ref>
</portlet>
</portlet-app>
```



Java™ Portlet 2.0 Tag Library (1)

`<%@ taglib uri="http://java.sun.com/portlet_2_0" prefix="portlet" %> :`

- `<portlet:defineObjects/>` - defines xxxRequest, xxxREsponse, portletConfig, portletSession, portletSessionScope (as Map), portletPreferences, portletPreferencesValues (as Map)
- `<portlet:actionURL copyCurrentRenderParameters="true" windowState="normal" portletMode="edit" name="setDeafaults">`
- `<portlet:renderURL portletMode="help" windowState="maximized">`
- `<portlet:resoureURL id="todos/todo/5" cacheability="PAGE">`
- `<portlet:param name="groupID" value="5"/>`



Java™ Portlet 2.0 Tag Library (2)

`<%@ taglib uri="http://java.sun.com/portlet_2_0" prefix="portlet" %> :`

- `<portlet:namespace/>` - defines unique namespace prefix for the id in the web page, JavaScript function names, etc.
- `<portlet:property name="important" value="true"/>` - sets properties to `<portlet:actionURL>`, `<portlet:renderURL>`, and `<portlet:resourceURL>`
- `escapeXML` attribute – defines if the url parameters, etc. should be escaped (true by default) - example:

`<container-runtime-option>`

`<name>javax.portlet.escapeXml</name><value>>false</value>`

`</container-runtime-option>`



НОВИ ВЪЗМОЖНОСТИ В Java™ Portlet 2.0 API: Events

```
<portlet>
  ...
  <supported-publishing-event>
    <name>city_changed</name>
  </supported-publishing-event>
</portlet>
<default-namespace>http://iproduct.org/myportlets
</default-namespace>
<event-definition>
  <name>city_changed</name>
  <value-type>java.lang.String</value-type>
</event-definition>
```



НОВИ ВЪЗМОЖНОСТИ В Java™ Portlet 2.0 API: Events

```
<portlet>
  ...
  <supported-processing-event>
    <name>city_changed</name>
  </supported-processing-event>
</portlet>
<default-namespace>http://iproduct.org/myportlets
</default-namespace>
<event-definition>
  <name>city_changed</name>
  <value-type>java.lang.String</value-type>
</event-definition>
```



НОВИ ВЪЗМОЖНОСТИ В Java™ Portlet 2.0 API: Events

- Publishing

```
actionResponse.setEvent("city_changed", city);
```

- Processing
@Override

```
public void processEvent(EventRequest req, EventResponse resp)
    throws PortletException, IOException {
    Event event = request.getEvent();
    if(event.getName().equals("city_changed")){
        String newCity = (String) event.getValue();
        response.setRenderParameter("city", newCity);
    }
```



Portlet 2.0 API: Public Render Parameters

```
<portlet>
  ...
  <supported-public-render-parameter>
    location
  </supported-public-render-parameter>
</portlet>

<public-render-parameter>
  <identifier>location</identifier>
  <qname xmlns:my="http://iproduct.org/myportlets">
    my:geolocation
  </qname>
</public-render-parameter>
```



Бъдещето: Web Services for Remote Portlets (WSRP)

- WSRP v1 е OASIS стандарт от September, 2003
- WSRP v2 е одобрен от OASIS на 1 април 2008
- WSRP v2 поддържа Web 2.0 технологии, като **AJAX** и **REST**
- Позволява портлетите да бъдат достъпвани отдалечено и по-лесно комбинирани от портали на трети страни
- Поддръжка от всички основни играчи на, включително **Oracle®**, **IBM®**, **Microsoft®**
- Крайната цел на WSRP е да направи достъпни предимствата на **Service-Oriented Architecture** за крайния потребител



JSR 314: JavaServer Faces 2.0 (1)

- За да може портлетната технология да стане още по успешна и широко използвана са необходими рамкови среди и инструменти за разработка (**developer frameworks and tools**), които поддържат скоростната разработка на портлетни приложения (**rapid application development**) и многократното използване (**reuse**) на компоненти
- Като част от Java™ Enterprise Edition процеса на стандартизация и като препоръчвана технология за уеб базирана презентация в последната версия на Java™ EE 6, **Java Server Faces (JSF) Model-View-Controller (MVC)** технологията е доста обещаващ кандидат за подобна **enabling technology**



JSR 314: JavaServer Faces 2.0 (2)

Сред предимствата на JSF са:

- лесно конструиране на UI чрез изграждане от множество многократно използваеми компоненти;
- ясно разделяне на данни и презентация с използване на MVC шаблона (design pattern);
- лесен за използване модел за свързване и обработка на събития генерирани от клиента към кода на приложението на сървъра;
- състоянието на UI компонентите се управлява автоматично между клиентските заявки;
- разделяне на отговорностите между корпоративните разработчици и системните програмисти.



JSR 314: JavaServer Faces 2.0 (3)

Последната версия **JSF 2.0** дава допълнителни предимства:

- по-гъвкави, базирани на стандарти презентационни компоненти с използване на **фейслети (facelets)**;
- лесен за използване механизъм за създаване / използване на **шаблони на страници (view/page templating)**;
- лесни за създаване **потребителски компоненти (custom components)** без нужда от писане на Java код чрез **композиране** на съществуващи компоненти;
- невидима за клиент-програмиста и **унифицирана интеграция** с всички различни типове **beans** (ManagedBeans, POJO, EJB) чрез използване на **dependency injection** анотации;



JSR 314: JavaServer Faces 2.0 (4)

Последната версия JSF 2.0 дава допълнителни предимства:

- нови обхвати на видимост (напр. **conversation scope**, **custom scopes**);
- **bookmarkable application states** чрез използване на **view parameters**;
- По-добра поддръжка на **ajax** с използване на стандартни тагове – няма нужда да пишем (и поддържаме) допълнителен JavaScript код;
- **partial view processing & rendering** по време на **ajax** заявките;
- по-лесни конфигуриране, навигация между страниците и зареждане на ресурси.



JavaServer Faces HTML Tags Reference - Mozilla Firefox


File Edit View History Bookmarks Tools Help

http://www.exadel.com/tutorial/jsf/jsftags-guide.html

jsf tags guide

liferay.com - IPT Vi... dataTable (ICEface... Data Table Compon... Tree Component tu... portletfaces.org - E... JavaServer ... Overview (Java EE ... Portlet API (V0.7.1) ProcessEvent

alt="jsr-sun" url="/images/jsf-sun.gif"></h:graphicImage>



src="/jsr-example/images/jsr-sun.gif" alt="jsf-sun" />

UIInput

inputText

<h:inputText id="address" value="#{jsfexample.address}" />

<input type="text" name="jsftags:_idl" value="123 JSF Ave" />

inputSecret

<h:inputSecret redisplay="false" value="#{jsfexample.password}" />

<input id="jsftags:password" type="password" name="jsftags:password" value="secret" />

inputHidden

<h:inputHidden id="hidden" value="userPreference" />

No Renderer

<input id="jsftags:hidden" type="hidden" name="jsftags:hidden" value="userPreference" />

inputTextArea

<h:inputTextArea id="textArea" rows="4" cols="7" value="Text goes here." />

Text goes here..

<textarea id="jsftags:textArea" name="jsftags:textArea" cols="5" rows="3">Text goes here

Find: Generic

Next Previous Highlight all Match case

Done

Start 4 Win... ajaxpor... 2 Fire... liferay... liferay... Tomcat 3 Ado... 2 Ope... 5 Micr... IPT_Por... EN

9:30 AM

JSR 329: Portlet 2.0 Bridge for JavaServer™ Faces 1.2 Specification (1)

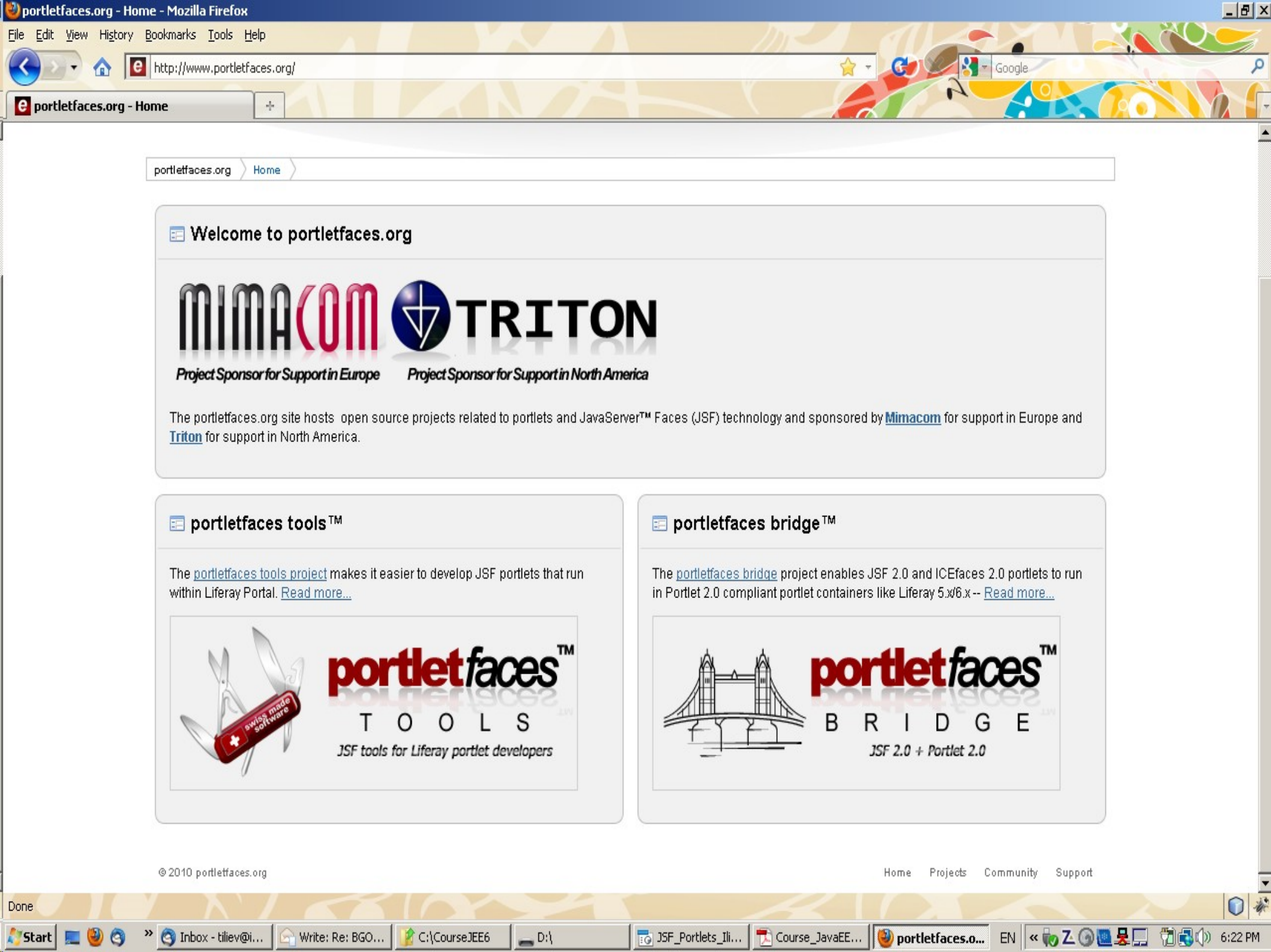
- JSF технологията осигурява много удобни възможности за опростяване на процеса на разработка, но има и някои фундаментални проблеми JSF да се използва директно за разработка на портлети – **Какъв е проблемът?**
- Най-същественият проблем е разликата в **жизнените цикли на портлетите и JSF компонентите** – портлетите имат отделни **action, event, resource и render requests**, докато JSF сървлетът обработва заявките следвайки стандартен цикъл на обработка (**request processing lifecycle**) заявка-отговор.



JSR 329: Portlet 2.0 Bridge for JavaServer™ Faces 1.2 Specification (1)

- Поради това е необходим мост (bridge) между тези две технологии за да можем да комбинираме техните предимства:
 - JSR 301: Portlet 1.0 Bridge for JavaServer™ Faces 1.2, и
 - JSR 329: Portlet 2.0 Bridge for JavaServer™ Faces 1.2 Specification.
 - А има ли JSR ??? : Portlet 2.0 Bridge for JavaServer™ Faces 2.0 Specification ???
 - Not yet officially :(...
 - BUT unofficially YES: PortletFaces.org, Apache MyFaces, JBoss Portlet Bridge, ... :)





Welcome to portletfaces.org



The portletfaces.org site hosts open source projects related to portlets and JavaServer™ Faces (JSF) technology and sponsored by [Mimacom](#) for support in Europe and [Triton](#) for support in North America.

portletfaces tools™

The [portletfaces tools project](#) makes it easier to develop JSF portlets that run within Liferay Portal. [Read more...](#)



portletfaces bridge™

The [portletfaces bridge project](#) enables JSF 2.0 and ICEfaces 2.0 portlets to run in Portlet 2.0 compliant portlet containers like Liferay 5.x/6.x -- [Read more...](#)



Last Published: 06 Apr 2011

JSR-329

JSR-301

Apache

MyFaces



Portlet Bridge

Overview

- Download
- Mailing Lists
- Issue Tracking
- Wiki Homepage

Projects

Portlet Bridges

- Version 3.0
- Version 2.0
- Version 1.0

TCK

- JSR-301
- JSR-329

Foundation

- ASF
- Sponsorship
- Thanks

Introduction

The MyFaces Portlet Bridge project provides implementations of the technology needed to expose a JSF application as a portlet within a Portlet 2.0 or Portlet 1.0 environment. This technology is defined by the Portlet Bridge for JavaServer Faces standards. This project provides the reference implementations for these standards and is being developed by the OpenSource community.

Though existing within the MyFaces project umbrella, these bridges don't require the use of the MyFaces implementation. They can be installed and run using any Faces environment. Likewise they are designed to run in any Portlet 2.0 (or Portlet 1.0) compliant portlet container.

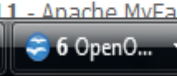
Currently the MyFaces Portlet Bridge project supports the following specifications:

- Portlet 2.0 Bridge for JavaServer Faces 2.0 - Currently a JSR hasn't yet been established for this version. Instead, because the work is overdue given that JSF 2.0 has been released for a while, this project is moving the JSR 329 bridge implementation forward to support and run in a JSF 2.0 platform outside the JSR process. In doing so it is preserving the JSR 329 apis while adapting and expressing the new JSF 2.0 features. Once the more immediate need of having a useful implementation is available, starting a new JSR will be looked at.
- **JSR-329**: Portlet 2.0 Bridge for JavaServer Faces 1.2 - This specification is final and has been approved. The corresponding Portlet 2.0 Bridge for JavaServer Faces 1.2 contained within this project is consistent with this specification and passes the TCK (Specification Test Conformance Kit). The TCK can also be found here and is for use by other implementations of JSR-329 that want to verify they conform with the final specification.
- **JSR-301**: Portlet 1.0 Bridge for JavaServer Faces 1.2 - This specification is final and has been approved. The corresponding Portlet 1.0 Bridge for JavaServer Faces 1.2 contained within this project is consistent with this specification and passes the TCK (Specification Test Conformance Kit). The TCK can also be found here and is for use by other implementations of JSR-301 that want to verify they conform with the final specification.

More information on these bridges and their supported specifications can be found on the [Apache MyFaces Portlet Bridge Projects Homepage](#) as well as the introduction pages of each of the listed bridge implementations.

News

- 04-05-11 - Apache MyFaces Portlet Bridge 3.0.0-alpha released!





Bridging today's most popular frameworks



RichFaces



Seam



GateIn

Overview

Downloads

Documentation

Community


Issue Tracker

Source Code

Build

About the Bridge

The JBoss Portlet Bridge is a non-final draft implementation of the JSR-301 and JSR-329 specifications to support JSF within a portlet and with added enhancements to support other web frameworks. Currently the bridge supports any combination of JSF, Seam, and RichFaces to run inside a portlet.

 **ANNOUNCEMENT:** 3.0.0.Beta1 Released! JSF2 and/or RichFaces 4!!!! [Hide Details](#)

Click here to [download](#). Or you can get started with this maven command:

```
mvn archetype:generate -DarchetypeCatalog=http://bit.ly/jbossportletbridge
```

*Note - You must build GateIn from trunk to run on JBoss AS6. Find the maven command in the [README.txt](#).



Ако искате да научите повече ...

- Заповядайте на курсовете които IPT провежда регулярно (<http://www.iproduct.org/>):
- Java™ Portlet Development with JSR 286: Portlet 2.0 API, Liferay & GateIn 3.1 – JSP™, JSF 2.0 & AJAX Portlets
- Архитектура ориентирана към услуги (SOA) и съвременни стандарти за моделиране на бизнес процеси (BPM) - SOAP, WSDL 2.0, REST, SOA, SCA, SDO, BPMN, WS-BPEL
- Web Programming with Java™ Technology: Servlet™ 3.0, JSP™, JSTL, EL, JSF 2.0, Facelets & Templating, AJAX, Comet
- Програмиране на езика Java™ - 3 модула





IPT - Intellectual Products & Technologies Ltd



Начало

Курсове

График

Бизнес услуги

Услуги - персонал

Клиенти

Стандарти

Проекти

Публикации

Контакти

Обновена: 21.09.2011

Курсове

• IT курсове:

- [Java™ Portlet Development with JSR 286: Portlet 2.0 API, Liferay & GateIn 3.1 – JSP™, JSF 2.0 & AJAX Portlets](#)
- [Advanced JavaScript – jQuery, Yahoo! UI 3, Dojo Toolkit, HTML 5, WebGL & Web 2.0 Mashups](#)
- [Програмиране на езика Java™](#)
- [Графични и мрежови приложения на езика JAVA™](#)
- [Разработка на разпределени и мрежови приложения на езика JAVA™](#)
- [Web Programming with Java™ Technology: Servlet™ 3.0, JSP™, JSTL, EL, JSF 2.0, Facelets & Templating, AJAX, Comet](#)
- [Java™ Enterprise Technologies \(Java™ EE 6\) – EJB™ 3.1, JSF 2.0, Web Services, JAX-RS, JAXB, JNDI™, JTA™, JMS™, JPA 2.0](#)
- [Архитектура ориентирана към услуги \(SOA\) и съвременни стандарти за моделиране на бизнес процеси \(BPM\) – SOAP, WSDL 2.0, REST, SOA, SCA, SDO, BPMN, BPDM, WS-BPEL](#)
- [Унифициран език за моделиране \(UML™\) – графична нотация и приложения](#)
- [Процес на разработка на софтуер с използване на UML™](#)
- [Extensible Markup Language \(XML\) технологии и приложения](#)

• Курсове за дизайнери – уеб-дизайн и мултимедия

- [Мултимедийни технологии](#)
- [Уеб дизайн с HTML 5](#)

• Курсове за начинаещи

- [Компютърна грамотност и Интернет](#)



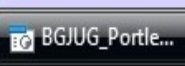
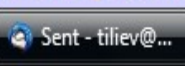
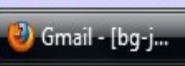
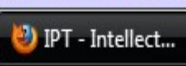
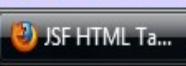
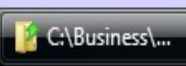
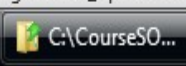
IT курсове:
Програмиране
на
JAVA™



- Core Java™ Programming
- Java™ APIs & GUI Design
- Network Computing

Java е търговска марка на Sun Microsystems, Inc.

www.iproduct.org





IPT - Intellectual Products & Technologies Ltd



Начало

Курсове

График

Бизнес услуги

Услуги - персонал

Клиенти

Стандарти

Проекти

Публикации

Контакти

Обновена: 21.09.2011

Java™ Portlet Development with JSR 286: Portlet 2.0 API & Liferay® – JSP™, Struts, JSF & AJAX Portlets (1 модул, 40 учебни часа)

In order to cope with the challenges of contemporary business environment including globalization, economical pressure for better efficiency, business process outsourcing, achievement of regulatory compliance, the enterprise needs improved support by technology. Recently the term Service Oriented Architecture (SOA) has become a pervasive buzz word together with *Cloud Computing*, *Software as a Service (SaaS)*, etc. The SOA standards like: *Simple Object Access Protocol (SOAP)*, *Web Services Description Language (WSDL)*, *Representational State Transfer (REST)*, *Web Application Description Language (WADL)* have emerged and matured. But from the end user perspective all these technologies do not matter much.

In 2002 the Java™ community started a pioneering effort for standardization of the web-based presentation services as well. They were called portlets and were able to be easily combined and integrated into enterprise portals based on a common standard defining their interaction – [JSR 168: Java Portlet Specification](#). In 2005 it was followed by another more advanced specification – [JSR 286: Portlet Specification 2.0](#).

Portals offer many advantages over other software applications. First, they provide a single access point for all employees, partners, and customers. Second, portals provide access to business functionality transparently from any device in virtually any location. Third, portals are highly flexible; they can exist in the form of B2E intra-nets, B2B extra-nets, or B2C inter-nets. Fourth, portals can be combined to form a portal network that can span a company's ecosystem. Fifth, because they provide front end for different web services they can easily integrate existing heterogeneous software systems and are future proof.

The Java™ Portlet technology (JSR 286: Portlet 2.0) allows easy sharing and combination of web applications developed by different organizations and individuals in a personalized enterprise portals. New "portlet-based" web application development style has emerged. A brief description for portlet in SOA (Service Oriented Architecture) style is: "presentation-oriented web service". *Portlet apps (PA)* are more distributed, flexible and agile, compared to older style, monolithic web applications we know for years. *PA* can use asynchronous data requests and can be dynamically updated in response (and sometimes in anticipation) to user's needs. They are typically consisting of several different portlets that communicate using shared parameters or publish/subscribe events according to latest *Portlet 2.0 Specification (JSR-286)*. [Liferay](#) is a state-of-the-art open-source Enterprise Portal Server licensed under LGPL, which was integrated by Sun in their portal server.

The IPT course *Java™ Portlet Development with JSR 286: Portlet 2.0 API & Liferay® – JSP™, Struts, JSF & AJAX Portlets* introduces to practical development of individual portlets, themes and layout templates, their integration in portlet applications, using the specific services provided by the portlet container, such as users' personal data management, web content management, etc., as well as use of tools and templates for efficient development of portlets (e.g. Liferay Plugins SDK). Among the unique opportunities provided by this course is the possibility to share the experience gained by IPT in building new lightweight "JavaScript portlets" which execute entirely on the client side, and their integration with the portal container provided services.

In order portlet technology to become even more successful are needed developer frameworks and tools that support rapid portlet application development and reuse of components. As part of Java™ Enterprise Edition standardization process and as a recommended technology for web based presentation in



IPT - Intellectual Products & Technologies Ltd



Начало

Курсове

График

Бизнес услуги

Услуги - персонал

Клиенти

Стандарти

Проекти

Публикации

Контакти

Обновена: 21.09.2011

- partial view processing and rendering during ajax requests;
- easier configuration, navigation and resource loading.

JSF technology provides many nice capabilities simplifying the development process, but there are some problems using JSF directly for portlet development. The most important one is the difference in lifecycles of portlets and JSF components – portlets separate action, event, resource and render requests, while standard JSF servlet handles them in a common request processing lifecycle. That is why a bridge between two technologies is needed in order to combine their advantages. Two new specification requests (targeted towards different versions of the portlet specification) were published to address this need – [JSR 301: Portlet 1.0 Bridge for JavaServer™ Faces 1.2](#), and [JSR 329: Portlet 2.0 Bridge for JavaServer™ Faces 1.2](#) Specification.

The course will present the **details of Java portlets development** using different web presentation technologies:

- [JavaServer Pages 2.2/Expression Language 2.2](#), [Standard Tag Library for JavaServer Pages \(JSTL\) 1.2](#), including [Asynchronous JavaScript + XML \(AJAX\)](#) portlet development
- [JSF 1.2](#) technology (including [JSR 301](#) and [JSR 329](#) open source implementations)

The course will also provide insight about emerging initiatives for development of [Portlet 2.0](#) to [JSF 2.0](#) bridge (no specification available yet) and practical development of portlets accessed through [Web Services for Remote Portlets \(WSRP\)](#) [OASIS](#) network protocol specification allows the portlets to be accessed remotely and combined easily by third party portals and consumers, supported by all of the portal market's major players, including Oracle®, IBM®, Microsoft®. The ultimate goal of WSRP is to bring the benefits of Service-Oriented Architecture to the end-user.

Practical examples using open source technologies will be demonstrated and developed by participants illustrating the concepts, and proving the value of the technologies. Participants will develop solutions to multiple problems and home work tasks with opportunity for individual consultation with the trainer. The implemented applications will be deployed on [Liferay 5 and 6](#) state-of-the-art open-source enterprise portal servers.

The course is led by Trayan Iliev – qualified trainer and university lecturer with 11-years pedagogical experience in Faculty of Mathematics and Informatics of Sofia University "ST. Kliment Ohridski" and IPT – Intellectual Products and Technologies. He has practical experience in technical development and management of multiple software projects (eLearning, WebTV, Web 2.0 Mashups, JavaScript Portlets).

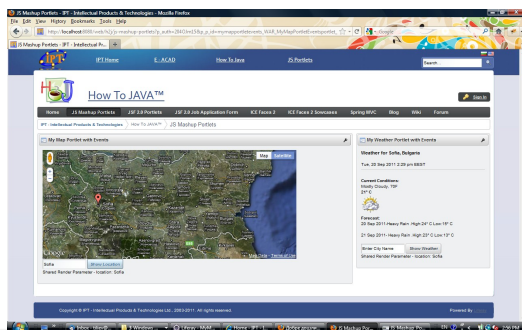
Ideally, the course participants should have previous knowledge and practical experience in Java Web programming technologies (JSP™, JSF, AJAX) that will be discussed during the course.

[Преглед на подробната програма на курса - PDF формат](#)

За повече информация и записване, моля пишете на нашия e-mail адрес: office@iproduct.org

Примерно приложение

Примерното **портлетно приложение** на тази страница демонстрира възможностите за **между-портлетна комуникация** с използване на **споделени параметри и събития** в JSR 286: Portlet 2.0. То се състои от два портлета интегриращи **JavaScript API: Google Maps 3 and Yahoo Weather** услуги. То представлява **mashup application** и демонстрира гъвкавостта при изграждането на mashups с Java™ Portlet технологията:



<http://www.h2j.org/bg/bgjug-portlets-demo>



Licensed under the **Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License**

Ресурси и референции (1)

- SOAP Version 1.2 Part 0: Primer (Second Edition) W3C Recommendation - <http://www.w3.org/TR/soap12-part0/>
- Web Services Description Language (WSDL) Version 2.0 Part 0: Primer - <http://www.w3.org/TR/2007/REC-wsdl20-primer-20070626/>
- Apache portals - <http://portals.apache.org/>
- OASIS Web Services for Remote Portlets (WSRP) - <http://www.oasis-open.org/committees/wsrp/>
- JSR 286: Java™ Portlet 2.0 API - <http://jcp.org/en/jsr/detail?id=286>



Ресурси и референции (2)

- JSR 314: JavaServer Faces 2.0 -
<http://www.jcp.org/en/jsr/detail?id=314>
- JSR 329: Portlet 2.0 Bridge for JavaServerTM Faces 1.2
Specification - <http://jcp.org/en/jsr/detail?id=329>
- Portlet 2.0 Bridge for JavaServerTM Faces 2.0 experimental
implementation - <http://www.portletfaces.org/>



Благодаря за вниманието!

Въпроси?

