Day4: ZK Introduction

Michael Mountrakis
mountrakis@illumineit.com

A training seminar given for [MOU S.A.](#)

Athens, December 2015

Day 4

1 ZK Introduction ←

2 ZK Design

3 ZK Implementation

ZK Introduction

Overview

- What is ZK
- Why we use it
- Client Server communication
- Designing ZK components
- Implementing ZK components

ZK vendor – Potix Corporation

- Vendor: Potix Corporation - Taiwan
  - http://www.zkoss.org/
  - Stakeholders: Acer Group, ASUS
- Introduced back in 2005
- Products Range:
  - ZK Framework, Web Visual Controls and Components over Java, AJAX
  - Opensource - SourceForge
  - Also commercialized regarding some components

ZK Framework

- Open Source Java Framework for building enterprise web and mobile apps
- First appearence: Version 1.0 February 2006
- Currently in version 8

ZK Framework - Aspects

- Write your application once and have it run anywhere
- Responsive design and components
- Resolution optimized automatically for web, mobile & tablets
- Only need to write once with one component set & one code-base
- Excellent user experience across desktops and devices

ZK Framework

- Why choosing it:
  - Opesource
  - Excellent look and feel
  - Hundrets of components
  - XML/AJAX Based
  - Java binding support
  - Large community
  - Excellent documentation and training
  - Millions of installations – widely used – matured framework.

ZK Framework and Java

- ZK is built on open source technologies such as
  - XML – ZUL extensions
  - AJAX
  - Java
  - jQuery
  - Bootstrap
- Also, applications build with ZK can be easily deployed on any Java Servlet or Application server.

ZK Intro

ZK Framework

- An AJAX framework based on JAVA
- XML UI Programming
- Pure Java Programming support
- Allow Fast Prototyping
- Strong MVC Support
- Eclipse, Netbeans IDE Support
- XHTML Compatible & UI Design

ZK Intro

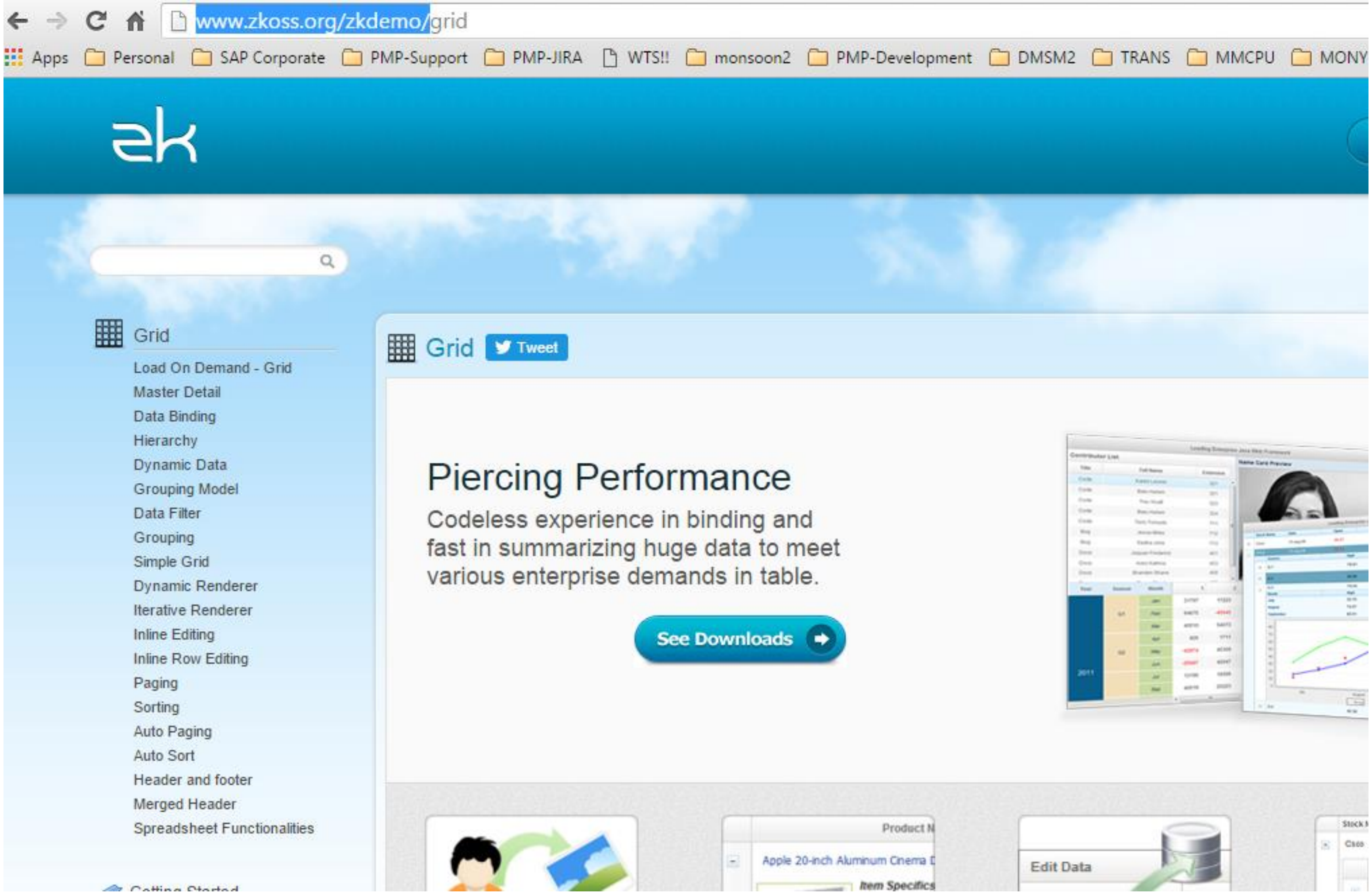
What is not ZK Framework

- No assumption to Persistence
- No assumption to middleware
- Does as thin as possible in controller layer
- Not limited in XUL
- Not limited in Browser

ZK Intro

Markup Language UI Programming

http://www.zkoss.org/zkdemo/



Piercing Performance

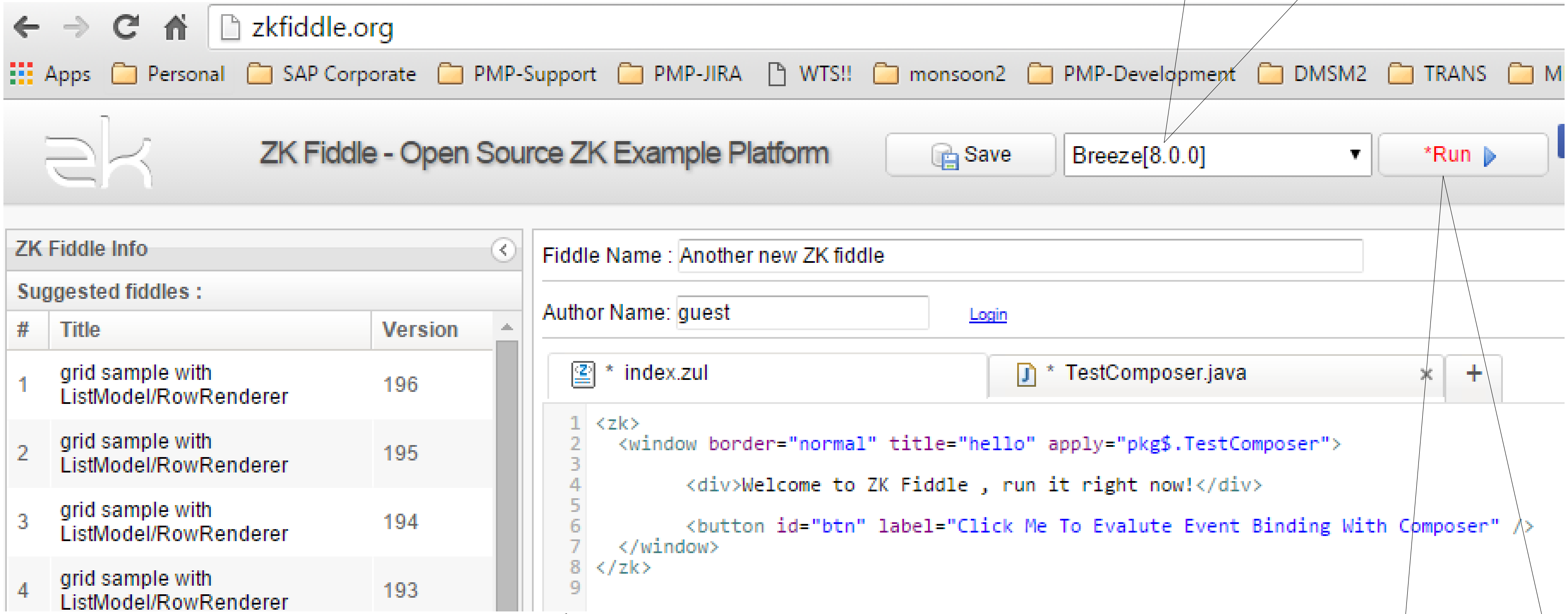
Codeless experience in binding and fast in summarizing huge data to meet various enterprise demands in table.

Markup Language UI Programming

- ZUML is ZK ZK User Interface Markup Language) is a markup language for rich user interfaces definition.
    - Designed for non-programmers to design user interfaces efficiently with the ZUML markup
    - allows developer to meld different markup languages, such as Mozilla XUL language and XHTML, seamlessly into the same page.
    - allows developers to embed script in pure Java language (interpreted by BeanShell) and use EL expressions to manipulate the components and access data.
    - ZUML is supported by ZK

Markup Language UI Programming

http://www.zkfiddle.org/



Fiddle Name : Another new ZK fiddle

Author Name: guest

Suggested fiddles :

#	Title	Version
1	grid sample with ListModel/RowRenderer	196
2	grid sample with ListModel/RowRenderer	195
3	grid sample with ListModel/RowRenderer	194
4	grid sample with ListModel/RowRenderer	193

```
<zk>
  <window border="normal" title="hello" apply="pkg$.TestComposer">

        <div>Welcome to ZK Fiddle , run it right now!</div>

        <button id="btn" label="Click Me To Evalute Event Binding With Composer" />
  </window>
</zk>
```

Check Compatibility

1 Write your ZUML code

2 Test it to see how it renders

Markup Language UI Programming

A simple example of ZUML

```
<zk>
  <window border="normal" title="hello" apply="pkg$.TestComposer">
        <div>Welcome to ZK</div>
        <button id="btn" label="Click Me To Evalute Event Binding" />
    <button label="MyButton" onClick='alert("Hello ZK World")'/>
  </window>
</zk>
```

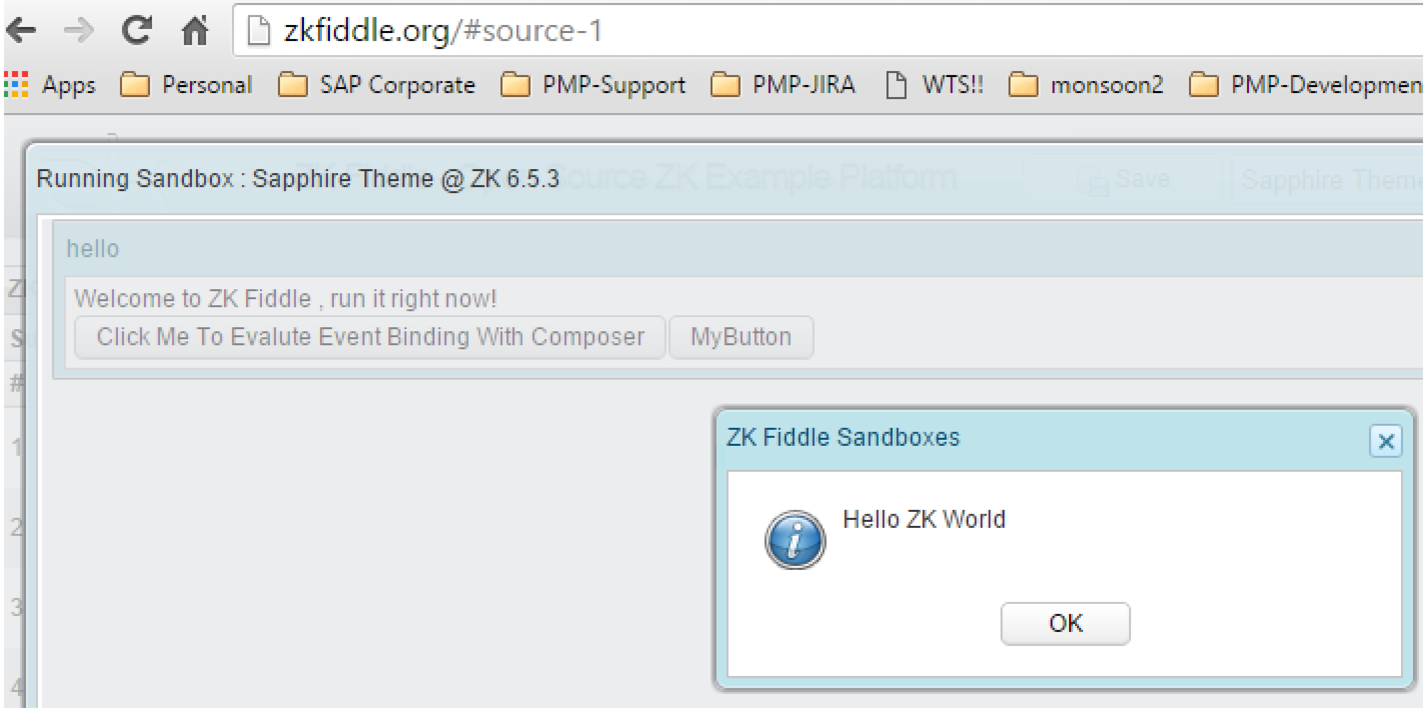
Add your button with an event
Handler for onClick() event.
Press *run*

ZK Intro

Markup Language UI Programming

A simple example of ZUML

```
<zk>
  <window border="normal" title="hello" apply="pkg$.TestComposer">
        <div>Welcome to ZK</div>
        <button id="btn" label="Click Me To Evalute Event Binding" />
    <button label="MyButton" onClick='alert("Hello ZK World")'/>
  </window>
</zk>
```



zkfiddle.org/#source-1

Running Sandbox : Sapphire Theme @ ZK 6.5.3

hello

Welcome to ZK Fiddle , run it right now!

ZK Fiddle Sandboxes

Hello ZK World

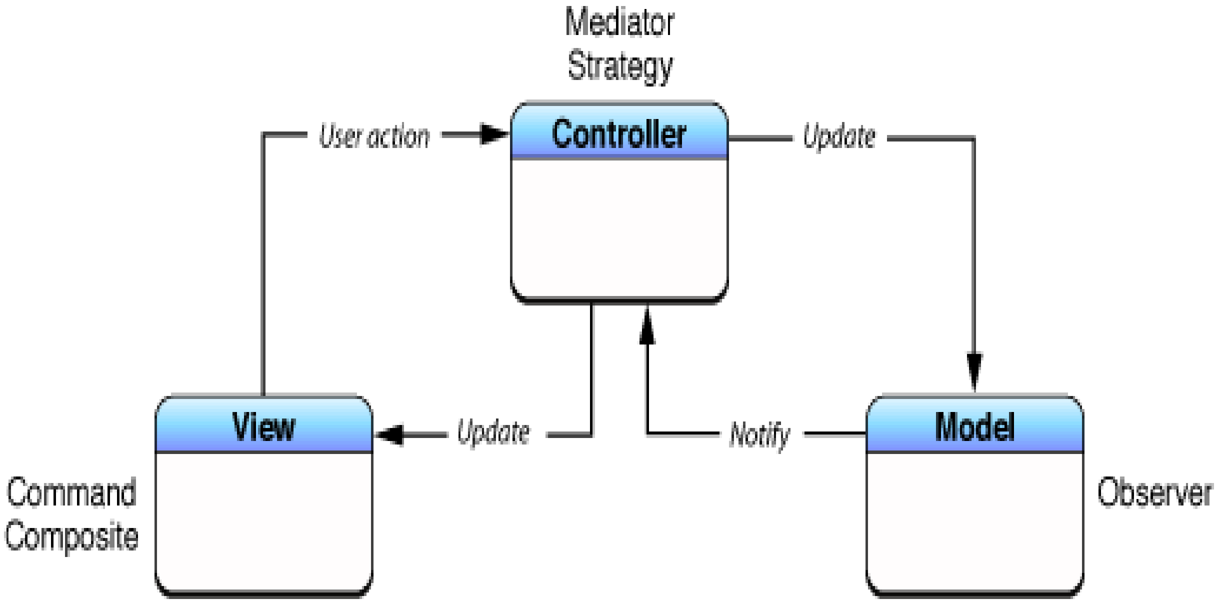
Markup Language UI Programming

- 200+ off-the-shelf state-of-art Ajax components
- Numerous third party widgets:
  - jQuery Plugin, JFreeChart, JasperReports, Google Maps, CKeditor …
- CSS-based skin, template-based look and customizable behavior
- Drag-and-drop, animation, context menu, bookmark management, …

ZK Intro

MVC Pattern

- MVC?
  - Model – View – Controller



MVC Pattern

- MVC?
  - Model – View – Controller
  - Divides a given software application into three interconnected parts, so as to separate internal representations of information from the ways that information is presented to or accepted from the use
  - Model: the data store/retrieve
  - View: output presentation, the UI
  - Controller
    - Controls Model to retrieve/refresh data
    - Signals View to change representation

MVC Pattern

- How ZK implements MVC?
- Model (Your Java Code)
  - anything you provide, from XML, JDBC, JPA,Spring,text....
- View (zul file)
  - The ZUML file for project presentation
- Controller (SelectorComposer)
  - The implementation in Java

ZK Intro

MVC Pattern

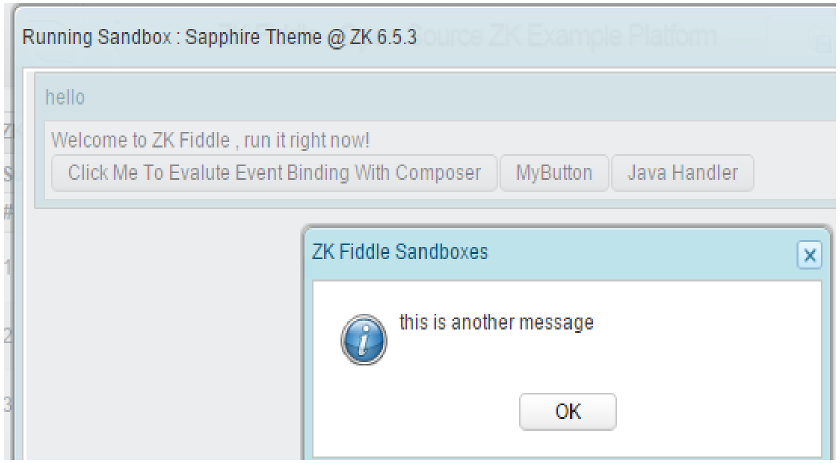
In the ZUML index.zul text add another button:

```
<zk>
  <window border="normal" title="hello" apply="pkg$.TestComposer">
       <div>Welcome to ZK Fiddle , run it right now!</div>
<button id="btn" label="Click Me To Evalute Event Binding With Composer" />
        <button id="btn2"  label="Java Handler"/>
  </window>
</zk>
```

Go to the TestComposer.java and add the method

```
public void onClick$btn2(Event e) throws InterruptedException{
       Messagebox.show("this is another message");
}
```

Pres Run



Running Sandbox : Sapphire Theme @ ZK 6.5.3

hello

Welcome to ZK Fiddle , run it right now!

ZK Fiddle Sandboxes

this is another message

UI Customization using CSS

- ZK utilizes pure CSS to customize visual controls and components
    - Pure CSS compatibility
    - Completely Style Customization

IDE Support

- ZK Framework supports the following IDE tools:
  - Eclipse: Maven build or ZK Studio
  - NetBeans: REM
  - IntelliJ: ZK Idea

Minimum Requirements

- Servlet Support: Containers from 2.3 and above
- JVM support: from Java 5 and above

Application Servers

Tomcat

Tomcat Cluster

JBoss

JBoss Cluster

WebSphere

WebSphere Portal

Glassfish Cluster

Oracle Server

WebLogic

WebLogic Portal

WebLogic Cluster

Google App Engine

Heroku

Red Hat OpenShift Express

Liferay

Pluto

Jetty

Resin

Information provided from:

http://books.zkoss.org/wiki/ZK%20Installation%20Guide

Programming ZK

- ZUML (ZK User Interface Markup Language) is based on XML. Similar to HTML and XUL, it is used to describe UI in an easy-to-understand format.
- In a ZUML document, each XML element instructs the ZK Loader which component to create.
- Each XML attribute describes what value to be assigned to the created component. Each XML processing instruction describes how to process the whole page, such as the page title.

Programming ZUL files with ZUML

Supported XML Instructions

- See http://books.zkoss.org/wiki/ZUML%20Reference/ZUML

- Import

– It imports a class or a package of classes. It works like Java's import statement.

- Link meta & script

– Link: It specifies an element that should be generated inside the HEAD element.

- Component

– Defines a new component based on a ZUML page. It is also called the macro component. In other words, once an instance of the new component is created, it creates child components based on the specified ZUML page

Programming ZUL files with ZUML

- Supported XML Instructions
- Xel-method
  - Specifies an EL function that could be used in EL expressions

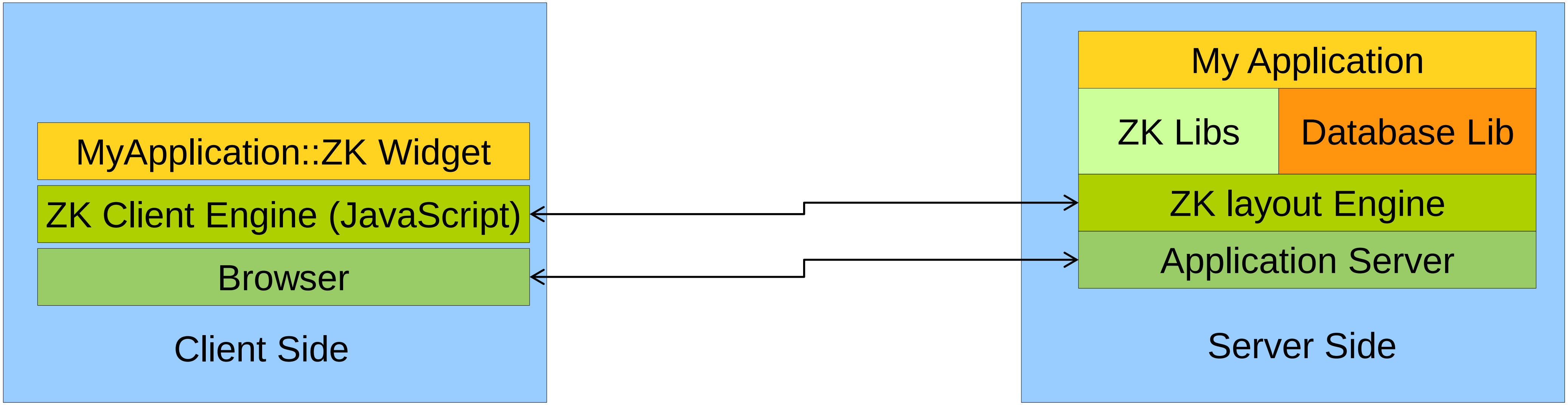
ZK Intro

Background

- Server + Client Fusion
- How ZK Works
  - How ZK Generates HTML page
    - ZUL Document Parsing
  - How ZK updates the page
    - ZK Page Update
  - Scope in ZK
  - Component Id space

Server Side + Client Fusion

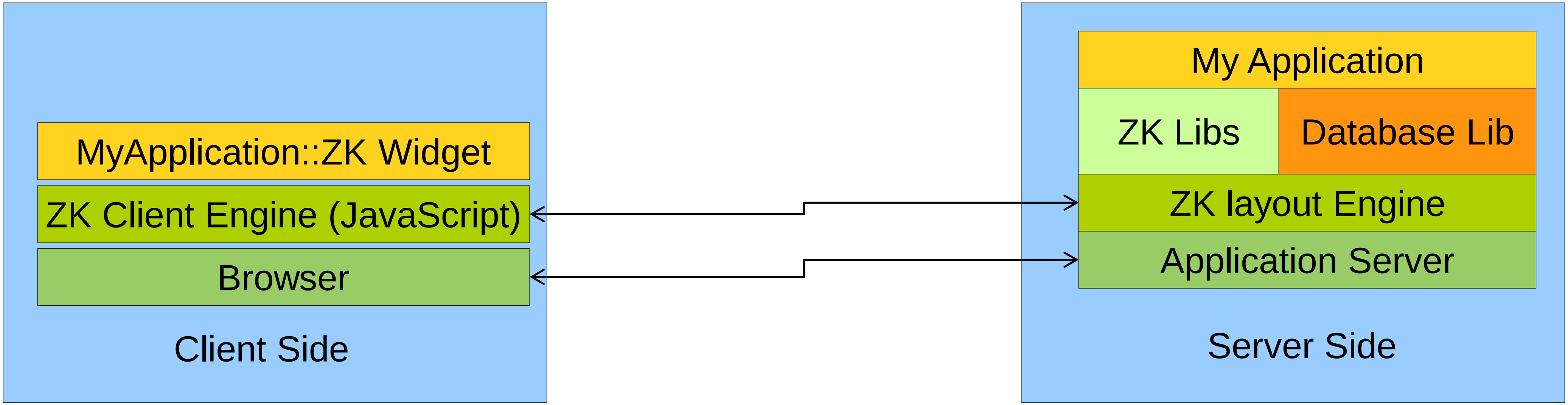
- ZK is Server Centric based architecture
  - All code runs on server side
  - Use of one language/technology to handle Model Logic, Business logic and presentation



ZK Intro

Server Side + Client Fusion

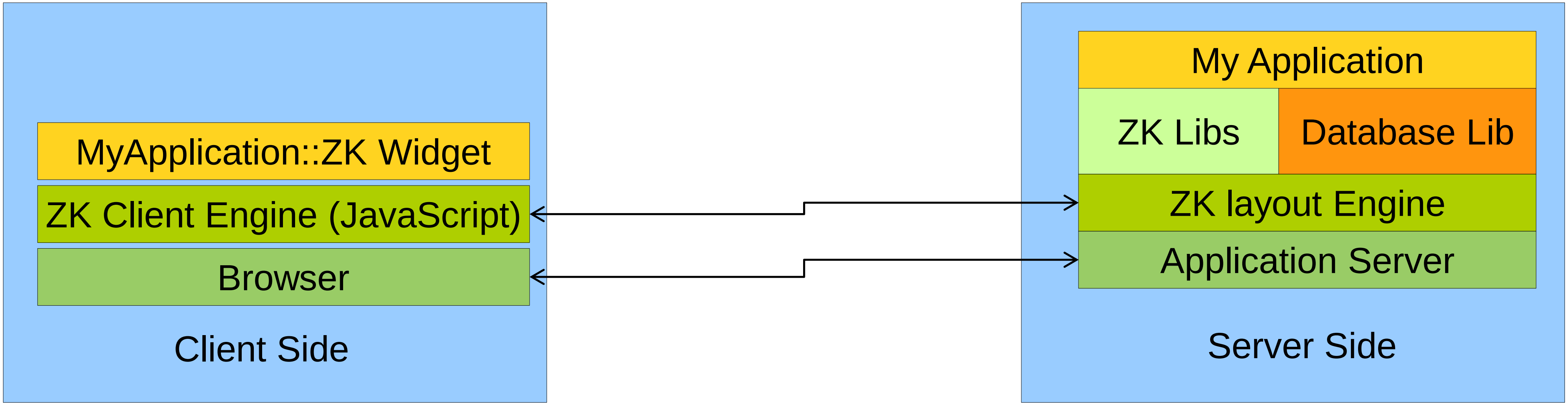
- Client Browser requests a URL/ZUL file
- Application Server receives the request, dispatches the request to ZK layout engine
- ZK Layout engine loads .ZUL file, parses and creates ZK component hierearchy described in it
- YOUR CODE RUNS NEXT <next slide>



ZK Intro

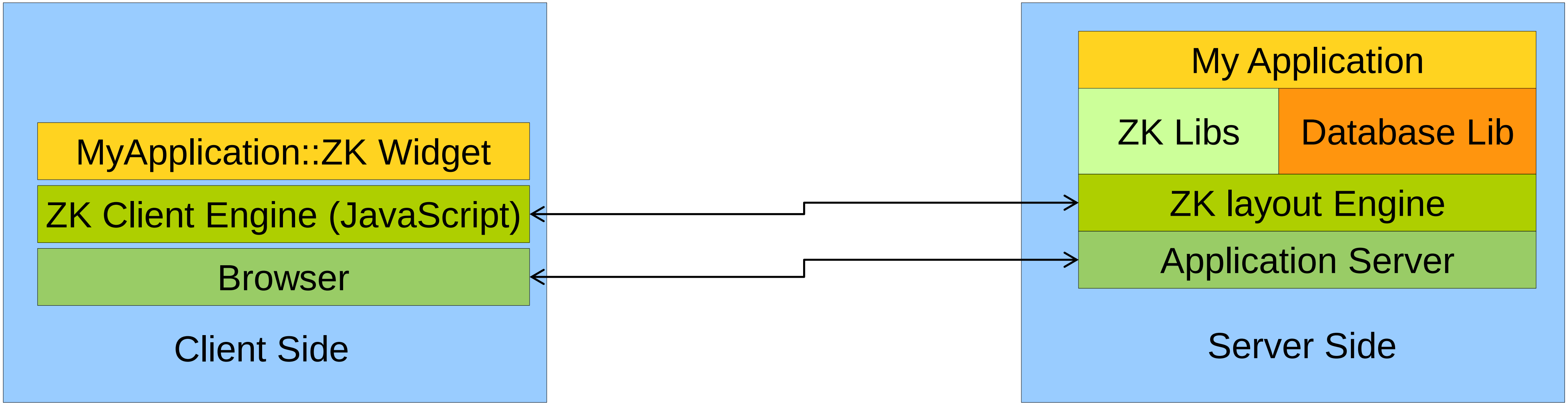
Server Side + Client Fusion

- YOUR CODE runs when the ZK Layout Engine initializes the Components you described in the ZUL file.
- Entry point in onCreate() of the Component
- Then all your Java code is traced: Database queries + web service cals
- ZK Components and controls of the ZUL file are bound to the data retirved by user´s code.



Server Side + Client Fusion

- Server/ZK Layout Engine response the created HTML page + the ZK client Engine which is written in JavaScript if not cached yet.
- Client Browser parses HTML+ ZK client engine javascript
- Client Browser renders HTML Page
- Client Browser runs ZK client engine javascript code



ZK Intro

ZUL Document Load

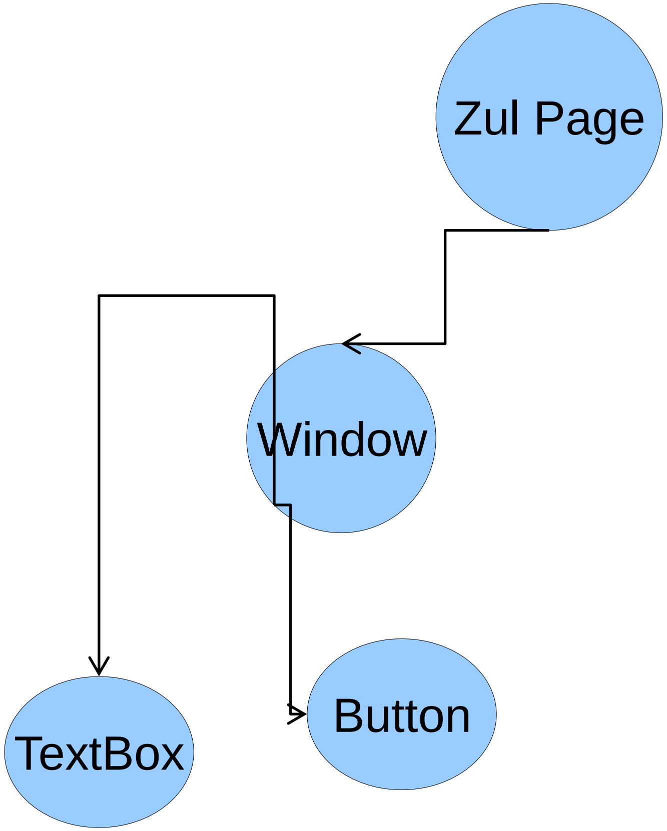
Follows the trace of ZK Layout Engine when it is requested to load a ZUL document. This is the loading cycle:

1.Page Initialization

2.Component Creation

3.Event Processing

4.Rendering



ZUL Document Load

Page Initialization

- OnCreate(): this is triggered when the component is created
- Init(): this is triggered after the page obect is created but none of the components of the page has been created
- AfterCompose(): is called in the Component Creation

ZUL Document Load

Component Creation

when components of the page are created.

- Order: from parent to children
- Order in page landscape: from top to bottom
- No argument constructor of each component
- Attributes of Components are populated by calling setter methods

ZUL Document Load

Event Processing:

In page loading phase, the onCreate event is the only event that will be handled unless user post other events inside onCreate event.

ZUL Document Load

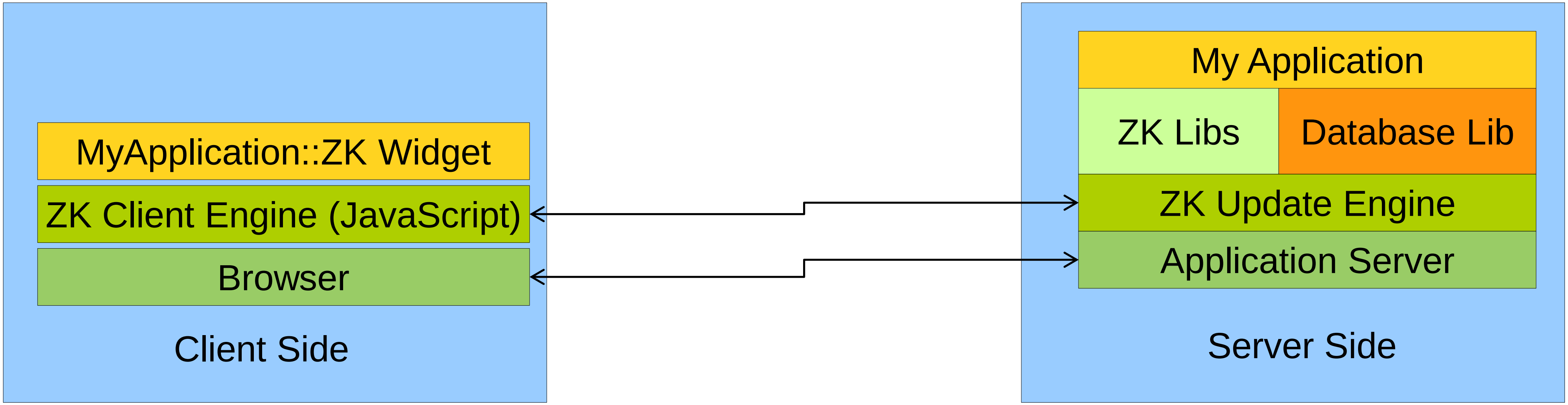
Rendering

- Follows Event Processing
- Component hierarchy is transformed to HTML code and JavaScript
- No Handler to application developer here. This is done by ZK engine

ZK Intro

ZUL Page Update

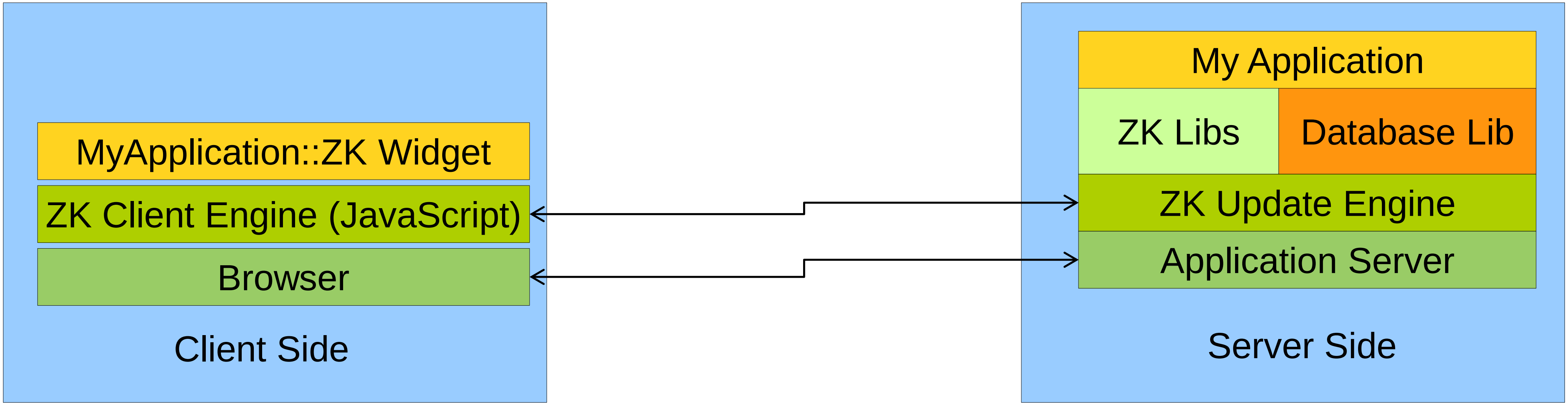
- User creates an event on Browser
- ZK Client engine sends it to server as AJAX event
- Event received from A/S dispatched to ZK Update Engine



ZK Intro

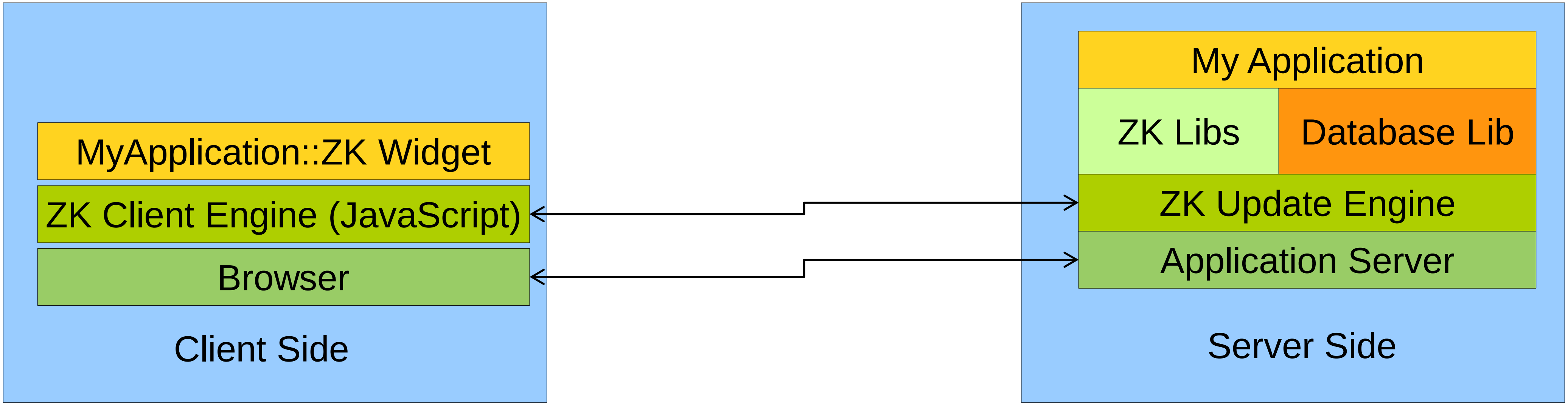
ZUL Page Update

- ZK Update engine receives the request
- ZK Update engine fires the corresponding event handler method, for example onClick()
- <YOUR CODE STARTS HERE>



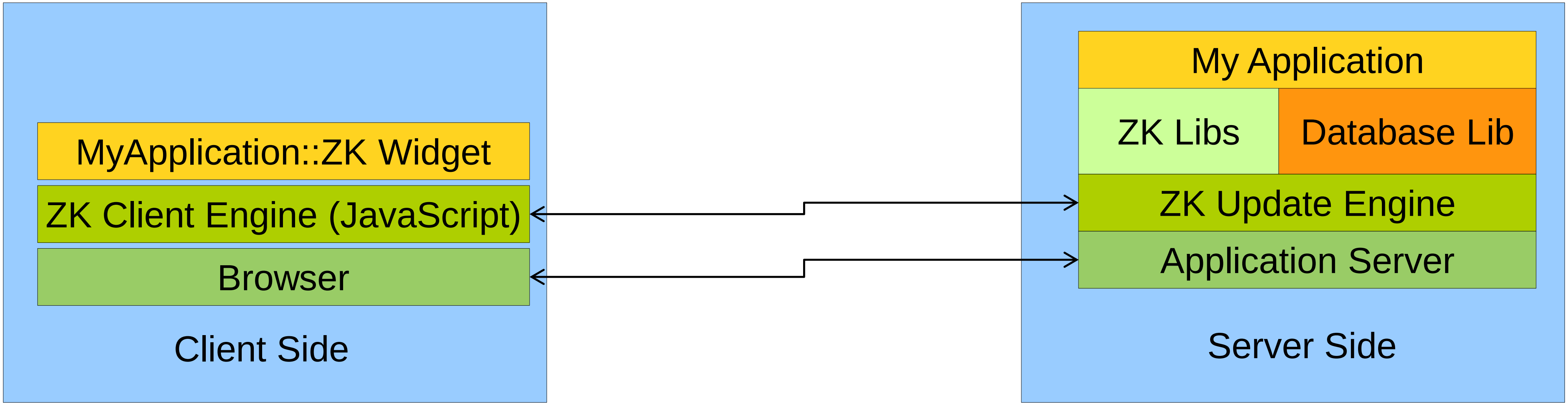
ZUL Page Update

- <YOUR CODE STARTS HERE>
- Connect to the database... call a service....
- Change the controls of the page
- <YOUR CODE STOPS HERE> --> Changes sent to ZK Update Engine



ZUL Page Update

- Changes sent to ZK Update Engine
- ZK detects updates, optimize them and send a response to the AJAX request
- Client: ZK Client engine get the response and transform the AJAX response to DOM changes so that the presentation of the application changes



ZK Page Update

- Request Processing
- Event Processing
- Rendering Phase

ZK Page Update

Request Processing

- User puts some text in a textbox
- Browser fires onChange() event
- ZK Client Engine catches onChange event and sends the onChange command to the Server ZK Update Engine in an AJAX message
- ZK Update Engine receives the AJAX message with the component id, the onChange event and submited new value
- ZK Update engine updates the textbox component and fires the method onChange of the component

ZK Page Update

Event Processing

- The Component´s (TextBox) onChange() method is triggered server side.
- Developer Code run on it.
- All subsequent events might be triggered from this one are dispatched one by one

ZK Page Update

Rendering Phase

- After all events originated from onChange() finish their execution
- The changes on Component state are detected from ZK Update Engine
- Changes optimized, packed to AJAX response and send to the client
- Client ZK Engine receives the request
- Client ZK parses the AJAX request and applies changes to the DOM of the presentation layer.

ZK Execution of Events

Client / Server AJAX message exchange

- All events from client are queued up in strict order
- No event is delivered to server in wrong order
- All events from server to client queued up in strict order
- No event is delivered to client in wrong order
- However....
- Application developer can handle them in a way he likes.

ZK Intro

ZK Page

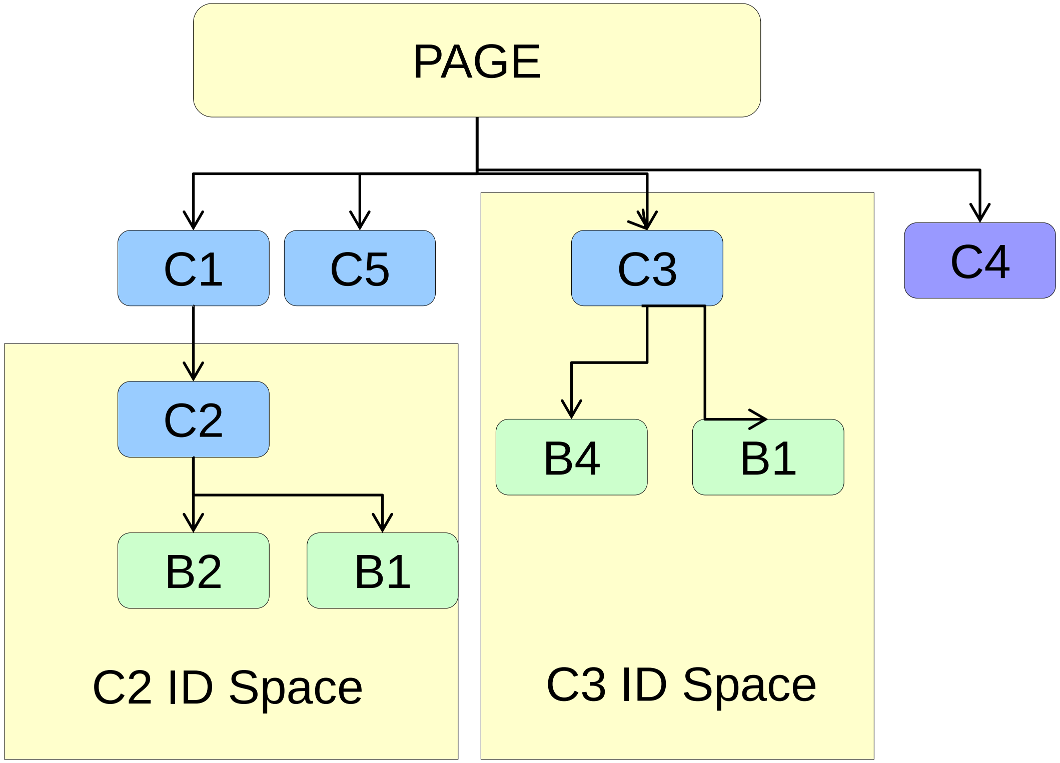
- Page
  - Is a collection of ZK components
  - Application programmer defines the placement of components in the page
  - Application programmer defines the visibility of components in the page
  - Page can have user defined attributes like id, title
  - Title is shown in the browser´s title bar
  - A page is mapped to a ZUL file.

Identification/Grouping of Components

- Each ZK Component has an *ID*
- This is the identification of the component
- Components can be grouped in an *ID space*
- Components can have the same ID but in different ID Spaces
- Components of the same *ID space* called *Fellows*
- Page and Window are the two default Id Spaces
- The root component that defines ID Space is called *space owner*

Identification/Grouping of Components

- C2 is Space Owner
- C3 is Space Owner
- Page is Space Owner
- B2/B3 are fellows in C2 ID space
- B1/B4 are fellows in C3 ID space
- C4 is Space owner without fellows.
- C1,C5are fellows in Page Id space



Resources

- ZK Documentation
  - http://www.zkoss.org/documentation
- ZK Forum
  - http://forum.zkoss.org/questions/
- ZK API Documentation http://www.zkoss.org/javadoc/

Any Questions?

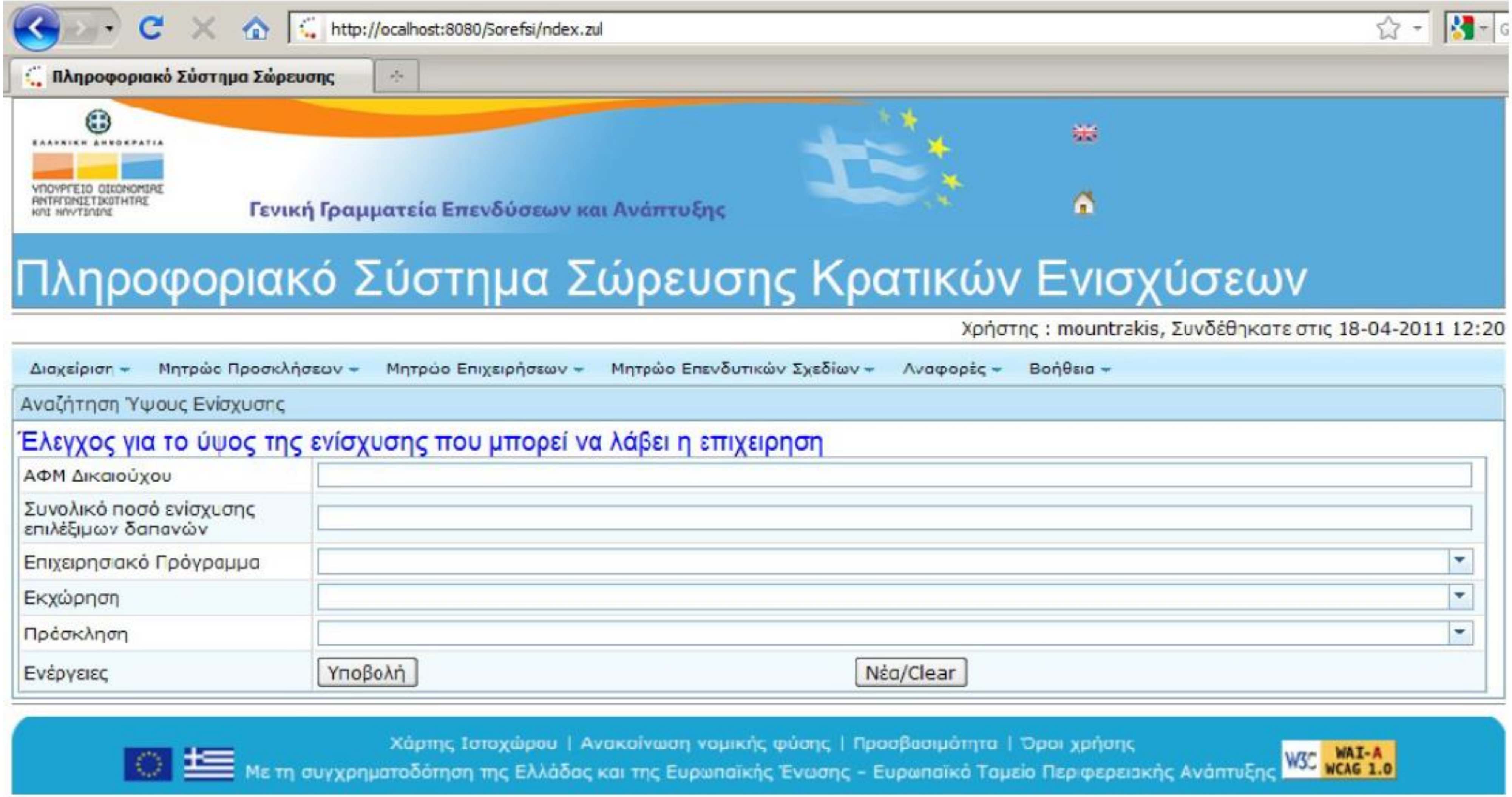
Lab

- Prepare Workstation
  - Java Version 6 or later from Oracle
  - Eclipse IDE for Java EE Developers - Mars
  - Configure Eclipse
    - To run with JDK rather than JRE
  - Setup Application Server: Apache Tomcat 7
    - Bind it with Eclipse
  - Install ZK Studio through Marketplace
  - See:

http://books.zkoss.org/wiki/ZK_Studio_Essentials/Installation

Lab

- Can you reverse engineer and reproduce the following GUI ?



http://ocalhost:8080/Sorefsi/ndex.zul
Πληροφοριακό Σύστημα Σώρευσης
ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΥΠΟΥΡΓΕΙΟ ΟΙΚΟΝΟΜΙΑΣ ΑΝΤΑΓΩΝΙΣΤΙΚΟΤΗΤΑΣ ΚΑΙ ΝΑΥΤΙΛΙΑΣ
Γενική Γραμματεία Επενδύσεων και Ανάπτυξης

Πληροφοριακό Σύστημα Σώρευσης Κρατικών Ενισχύσεων

Χρήστης : mountrakis, Συνδέθηκατε στις 18-04-2011 12:20

Διαχείριση ▾
Μητρώο Προσκλήσεων ▾
Μητρώο Επιχειρήσεων ▾
Μητρώο Επενδυτικών Σχεδίων ▾
Αναφορές ▾
Βοήθεια ▾

Αναζήτηση Υψους Ενίσχυσης

Έλεγχος για το ύψος της ενίσχυσης που μπορεί να λάβει η επιχειρηση

ΑΦΜ Δικαιούχου	
Συνολικό ποσό ενίσχυσης επιλέξιμων δαπανών	
Επιχειρησιακό Πρόγραμμα	▾
Εκχώρηση	▾
Πρόσκληση	▾
Ενέργειες	Υποβολή Νέα/Clear

Χάρτης Ιστοχώρου | Ανακοίνωση νομικής φύσης | Προσβασιμότητα | Όροι χρήσης

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης - Ευρωπαϊκό Ταμείο Περιφερειακής Ανάπτυξης

Lab

- Design a UI for a library with the following entities:
    - Library Subscriber
    - Book
    - Design the UI in paper
    - Design the ZUL files with ZUML in ZK Studio
    - Decide on Components to use per page
    - Decide on Events to be used per component
    - Set the events in ZUL

Lab

- View your design in tomcat
- Go to http://www.zkfiddle.org/
  - Test each of the componets
  - Try to define the TestComposer in java