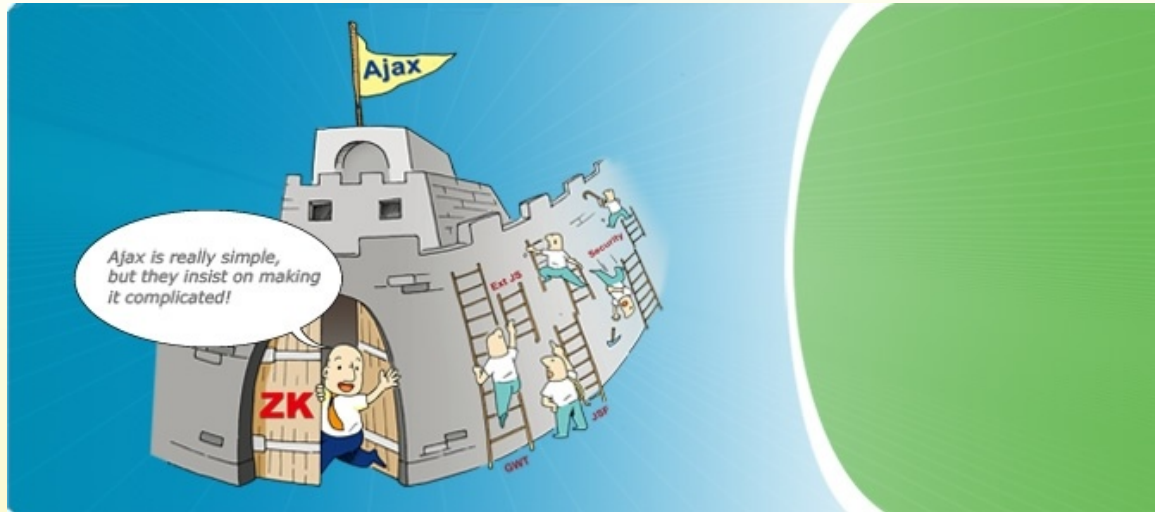




RIA mit ZK

Boost your productivity

Daniel Seiler, AIA 2008, Mainz

Agenda

- Introduction
- ZK basics
- ZK component library
- We build an application
- ZK advanced concepts
- Custom component example
- Integration example, Gmaps
- Summary

Goals of this session

Infect you with the ZK virus

You are able to explain the position of ZK
in the current RIA Landscape

You know the main features, concepts
and principles of ZK

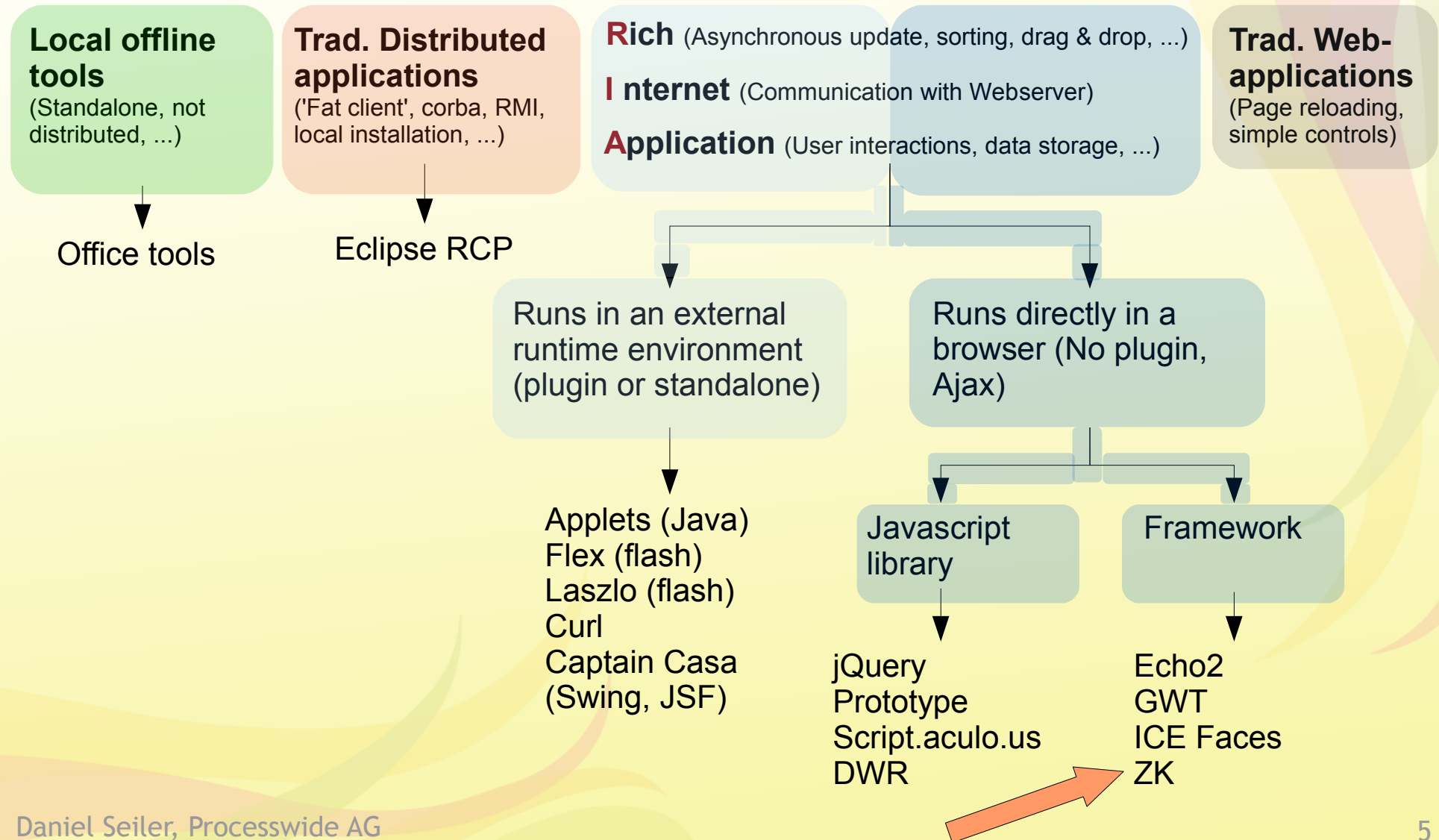
The problem to solve

To build rich, interactive, fast and scalable, distributed business applications ...

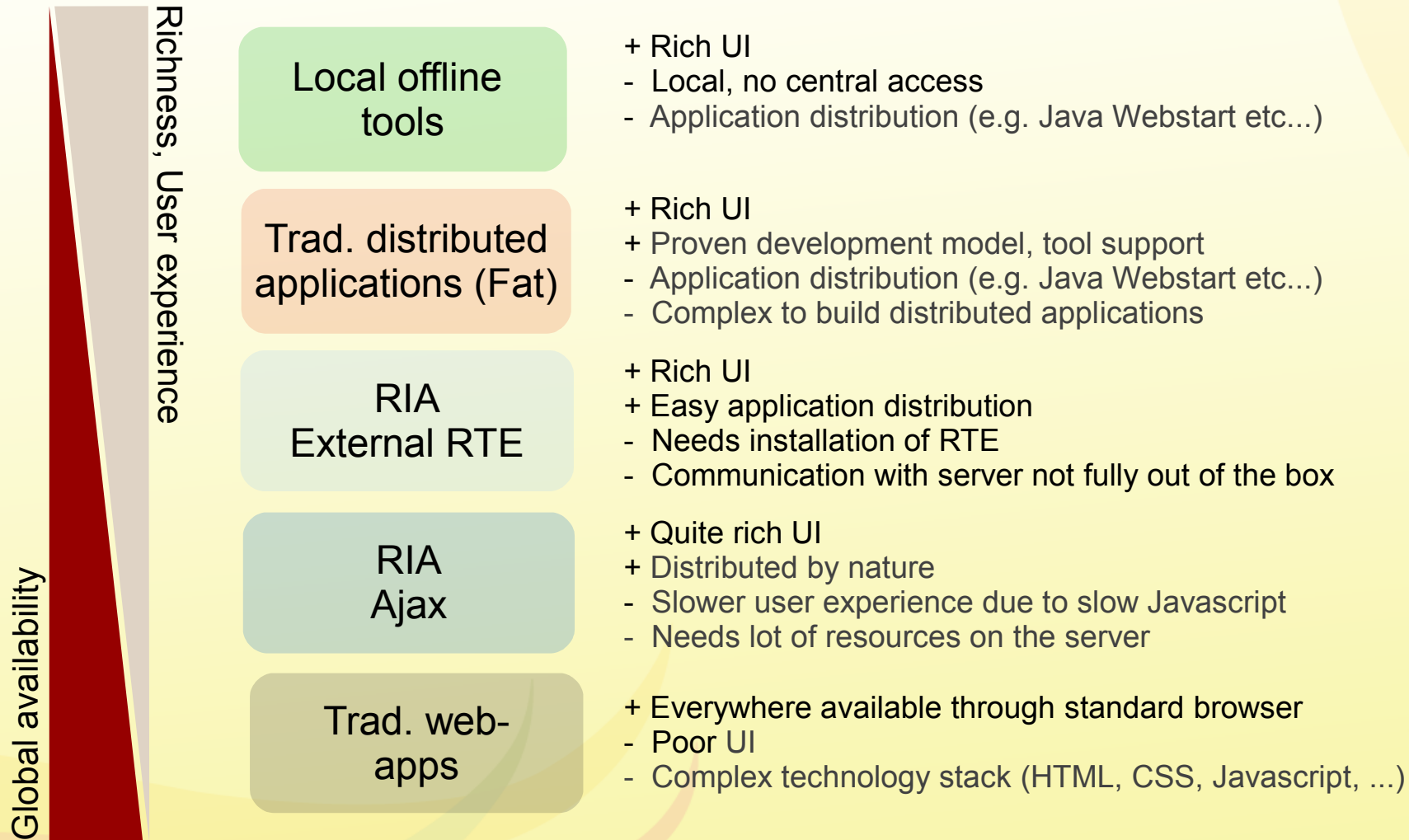
... we need a framework and technology that ...

- ... maximizes our productivity by abstracting and hiding much of the complexity
- ... provides a rich set of prebuilt components and features
- ... is easy to extend

The big picture



The right tool for your job



Why Ajax?

Basic idea:

breaking up the concept of the webpage to allow dynamic updates of certain areas within one page to increase the overall user experience

Typical technology stack:

- XHTML and CSS, Document Object Model, XML and XSLT
- The XMLHttpRequest object for asynchronous communication
- JavaScript to bring these technologies together

Disadvantages:

- Traditional webconcepts like back button and bookmarks need to be rethought
- Indexing through search engines is not guaranteed
- Depends on Javascript switched on in the browser
- Complex technology stack
- Javascript is quite slow. There is hope: (<http://www.google.com/chrome>)

Ajax tools

Snippet

jQuery, prototype, script.aculo.us, DWR



Widget

Yahoo UI Library, jbossrichfaces former Ajax4JSF, Dojo Toolkit



Framework

Backbase, ZK, Echo2, GWT, ICE Faces



What is ZK?

Web: <http://www.zkoss.org>

Developed by: **Potix Corporation**

11F-2, No.87, Zhengzhou Road
Taipei, 103
Taiwan, Republic of China
+886-2-2552-1002



According to wikipedia: http://en.wikipedia.org/wiki/ZK_Framework

ZK is an **open-source Ajax Web application framework**, written in Java, that enables creation of **rich graphical user interfaces** for Web applications with **no JavaScript** and little programming knowledge.

ZK takes the so called **server-centric approach** that the content synchronization of components and the event pipelining between clients and servers are automatically done by the engine and **Ajax plumbing codes are completely transparent** to web application developers.

Selling points

Open Source: ZK is the leading open source Ajax + Mobile framework. ZK developer community is extremely active with 20+ translations, 100+ articles/blogs, and 100,000+ lines of codes, 700,000+ downloads, from 190+ countries.

Rich User Experience: 170+ off-the-shelf state-of-art XUL/HTML-complaint Ajax components. Numerous third party widgets: JFreeChart, JasperReports, Google Maps, FCKeditor, Timeline, Timeplot, ExtJS, Dojo and so on

Server push support: Comet and Client polling

Standards-based: ZK is a standard-compliant solution.

Extensibility and Customizability: ZK is fully customizable and extensible with modular and plug-and-play architecture.

Mobile Access: ZK extends the reach of enterprise Internet applications to 1+ billion mobile devices at minimal cost. ZK supports Java Mobile, Android, and various mobile browsers.

Security: ZK is designed from the ground up to be secure.

Sun is using it for their virtual platform management suite:

<http://www.openxvm.org/xvmsui.html>

ZK Products

Framework

ZK Framework

ZK Mobile

Application component

ZK Spreadsheet

Development

ZK Studio

Enterprise Integration

ZK JSP, JSF, Portlet, RSS

ZK Forge

Integrating third party widgets, such as ZK Gmaps, ZK FCKeditor, ZK Timeline, ZK Ext JS, ZK Dojo and so on



Many Documents

JavaScript



Firefox

One Browser

One ZUL

```
<label value="Move Mouse Over Me!"  
      tooltip="edit"/>  
</window>
```

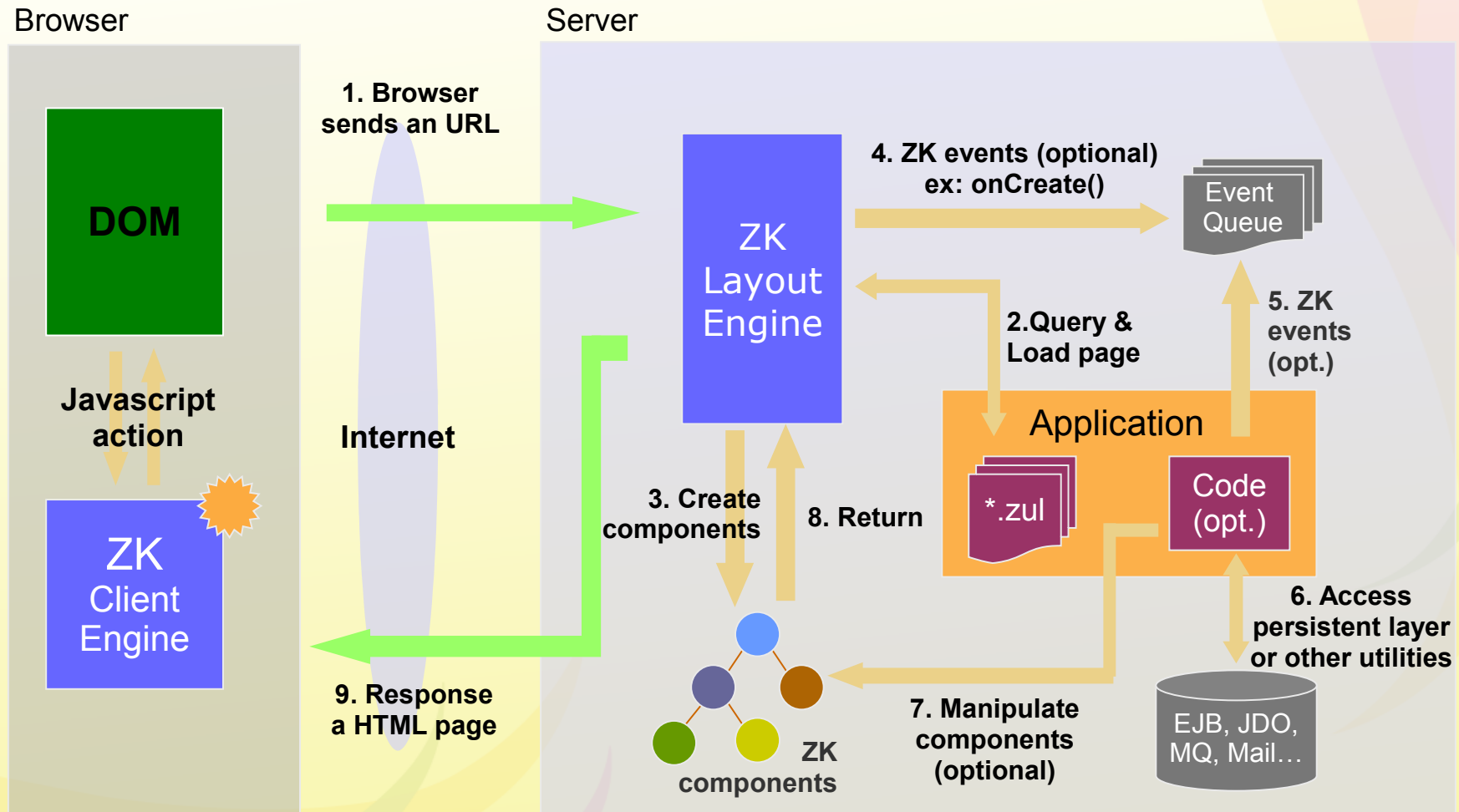
ZK Ajax Framework

IE7

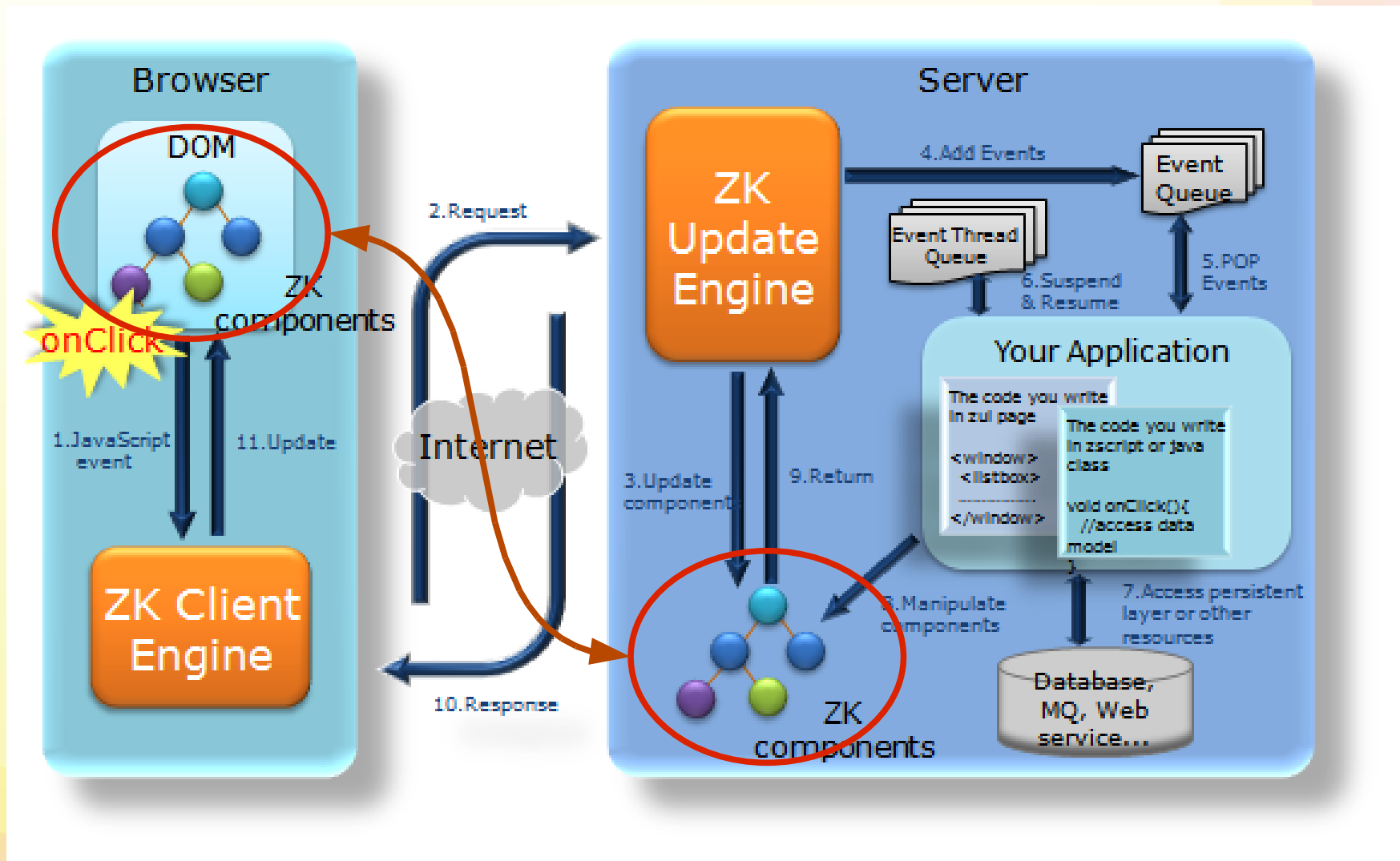
● ● ●

Many Browsers

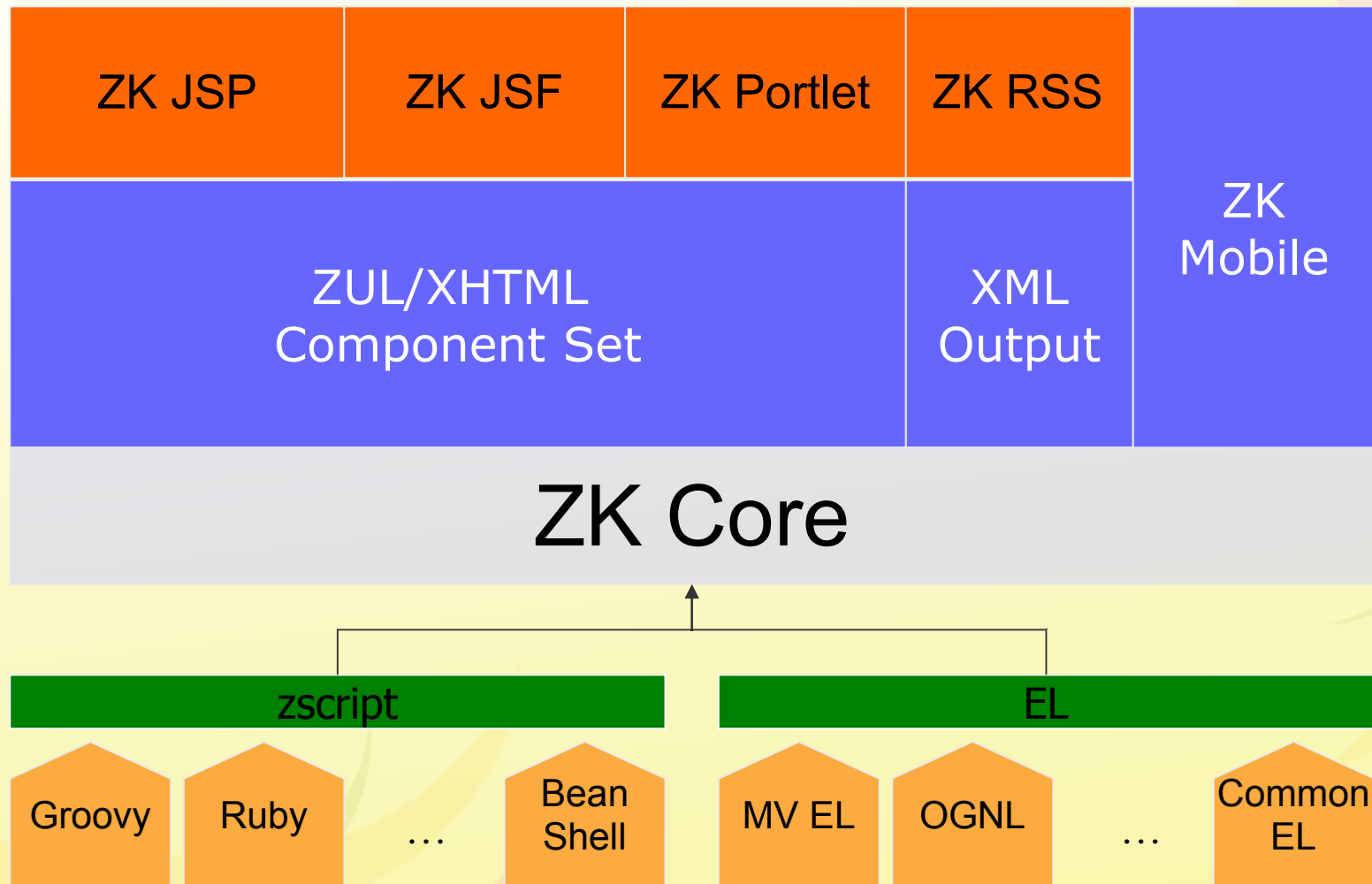
Generating an HTML page



ZK doing Ajax



ZK - Overall picture



ZK Components

See: <http://www.zkoss.org/zkdemo/userguide/>

- XUL (Mozilla) compliant
- 170+ off-the-shelf components
 - grid, tabbox, listbox, combobox, chart, splitter, slider, groupbox...
 - Context Menu
 - Drag and Drop
 - Live data (listbox and grid)
 - Auto-completion (combobox)
 - Customizable sorting (listbox and grid)

ZUL, EL and zscript

```
<window id="myWindow" title="fileupload demo" border="normal">
  <button label="Upload for ${myWindow.title}">
    <attribute name="onClick">
      {
        Object media = Fileupload.get();
        if (media instanceof org.zkoss.image.Image) {
          Image image = new Image();
          image.setContent(media);
          image.setParent(pics);
        } else if (media != null)
          MessageBox.show("Not an image:" + media, "Error",
            MessageBox.OK, MessageBox.ERROR);
      }
    </attribute>
  </button>
  <zscript>myWindow.appendChild(new Button("New Button"));</zscript>
  <vbox id="pics" />
</window>
```

Lets try it out on the ZK components Demo page ...

Our first application

Simple ZK Application - Microsoft Internet Explorer

Adresse: <http://localhost:8080/zkdemo/simple.zul>

Simple ZK Demo - (C) 2008 Processwide AG

Firstname*:

Lastname*:

Address:

Weight:

Birthdate:

Favorite color:

Married: ☒ Are you married?

How much do you like yourself?

☐ I hate myself ☐ Not so much ☐ I don't care ☒ I'm pretty neat ☐ I love myself

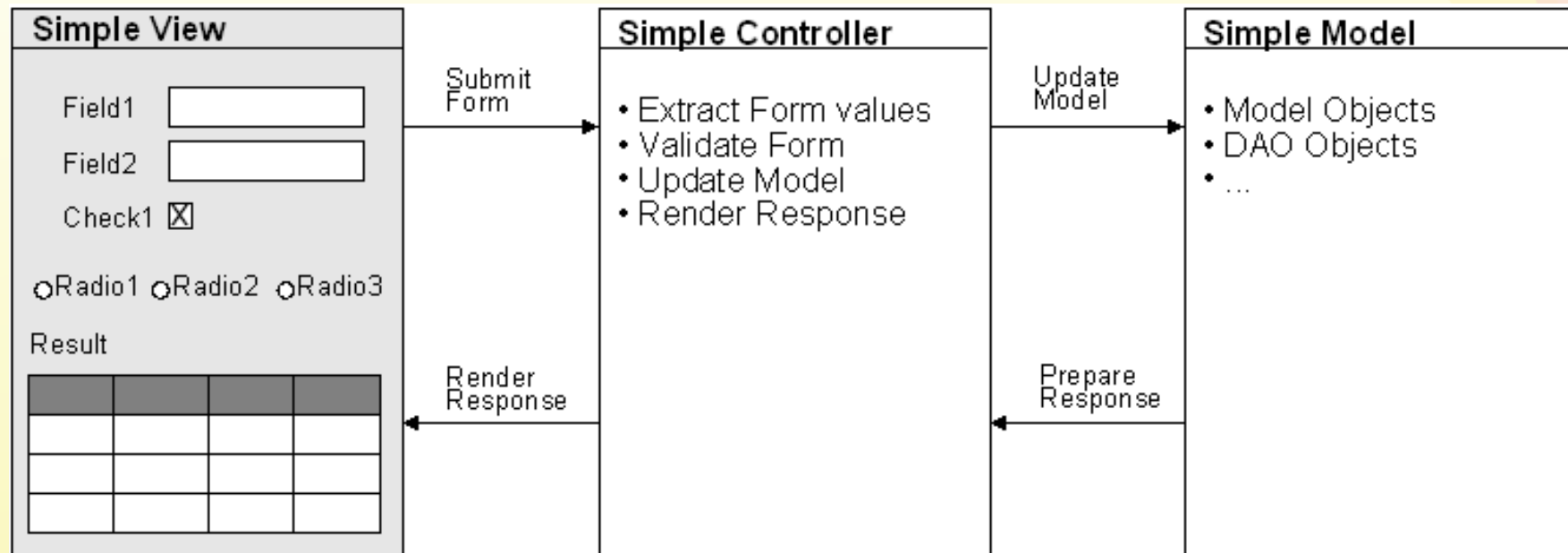
First Name	Last Name	Address	Weight	Birthdate	Color	Married	Rating
Daniel	Seiler	Bächlerweg 31	79.99	15.05.1973	Blue	☒	high

1) User enters his personal data

2) Data gets sent to the server and stored in a DB

3) Table gets updated with the new user

Architecture



simple.zul

```
<?page id="simplePage" title="Simple ZK Application"?>
<zkg>
<window id="simpleWindow" use="com.processwide.demo.zk.window.SimpleWindow"
  title="Simple ZK Demo" border="normal" width="800px" height="500px">
  <caption label="${simpleWindow.myCaption}" />
  <grid>
    <rows>
      <row>Firstname*:<textbox id="firstname" width="300px" /></row>
      ...
    </rows>
  </grid>
  ...
  <button label="Save" onClick="simpleWindow.addPerson()" />
  <grid id="personsGrid">
    ...
  </grid>
</window>
</zkg>
```

Declaration of
external Java class
as view handler

Expression language,
default: common-el

Scripting code,
default: BeanShell

Placeholder for dynamically
updated result table

SimpleWindow.java 1/2

```
public class SimpleWindow extends Window {
    private Grid personsGrid;

    public void onCreate() {
        createPersonsGrid();
    }

    public String getMyCaption() {
        return "(C) 2008 by Processwide AG";
    }

    private void createPersonsGrid() {
        personsGrid = (Grid)Path.getComponent("/simpleWindow/personsGrid");
        // add all the persons
        List<Person> persons = PersonDAO.getInstance().getAllPersons();
        for (Iterator<Person> it = persons.iterator(); it.hasNext();) {
            Person person = it.next();
            addPersonRecord(person);
        }
    }
    ...
}
```

SimpleWindow.java 2/2

```
...
public void addPerson() {
    // extracting the person data
    Textbox firstName = (Textbox) Path.getComponent("/simpleWindow/firstname");
    ...
    Person personBean = new Person();
    personBean.setFirstname(firstName.getValue());
    ...
    PersonDAO.getInstance().addPerson(personBean);
    addPersonRecord(personBean);
}
```

```
public void addPersonRecord(Person person) {
    Row row = new Row();
    Rows rows = personsGrid.getRows();
    rows.appendChild(row);
    row.appendChild(new Label(person.getFirstname()));
    ...
}
```

Add a delete button

```
public void addPersonRecord(final Person person) {  
    final Row row = new Row();  
    final Rows rows = personsGrid.getRows();  
    rows.appendChild(row);  
  
    // create the delete button  
    Button deleteBtn = new Button("delete");  
    deleteBtn.addEventListener("onClick", new EventListener() {  
        public void onEvent(final Event arg0) throws Exception {  
            // delete the person from the database  
            PersonDAO.getInstance().deletePerson(person.getId());  
            // update the table  
            rows.removeChild(row);  
        }  
    });  
    row.appendChild(deleteBtn);  
    ...  
}
```

Advanced concepts

- Data binding

```
<textbox id="firstname" value="@{person.firstname}"  
width="300px"/>
```

- Enhanced MVC

```
<window id="simpleWindow2" ...  
  apply="com.processwide.demo.zk.controller.SimpleComposer"  
  .. >
```

```
public class SimpleComposer extends GenericForwardComposer {  
  public void onClick$saveBtn(Event evt) {  
    Person person = (Person)page.getVariable("person");  
    ...  
  }  
}
```

- Custom components

ZK custom components

The ability to implement custom components easily and intuitively is one of the reasons for the productivity gain ZK brings compared to other frameworks

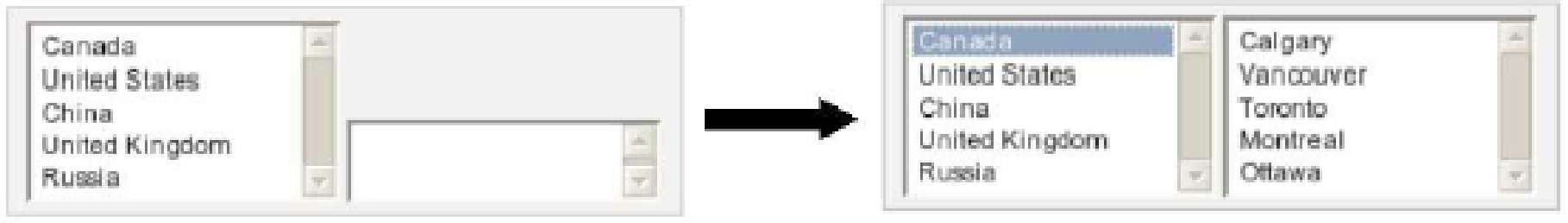
Create a custom component by ...

```
graph TD; A[Create a custom component by ...] --> B[Extending existing Components]; A --> C[Creating class deriving from org.zkoss.zk.ui.AbstractComponent];
```

Extending existing Components

Creating class deriving from
`org.zkoss.zk.ui.
AbstractComponent`

Double Combo



Client Solution

All the possible values are loaded at page loading to the client and changes are handled on the client through Javascript

- + Once loaded it is fast
- For large data sets the initial loading becomes impractical

Server Solution (see <http://www.ch.ch/karte/index.html?lang=de>)

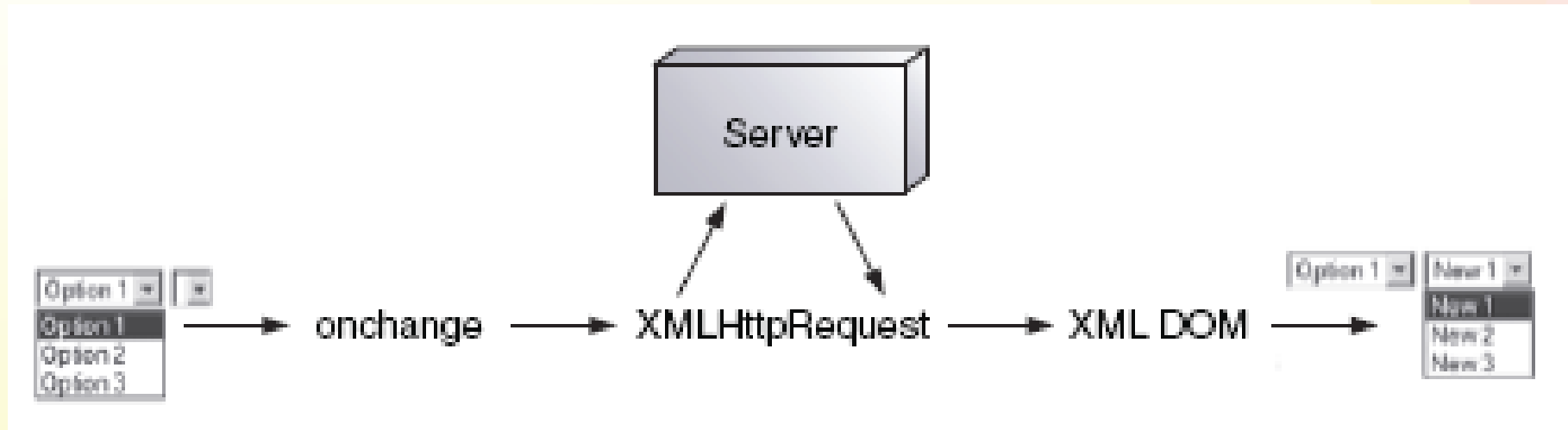
Each time the selection in the first combo changes, the whole page gets reloaded with the corresponding values filled into the second combo

- + Can handle large data sets
- The user experience, response time is bad due to reloading of the whole page

Ajax Solution

Each time the selection changes in combo1 an Ajax request is made and the combo2 gets updated without reloading the whole page

Pure Ajax solution



- + can handle large data sets
- + user experience and response time is good due to partial update of the page
- complex to implement without a framework like ZK !!

On chapter 9 of the book 'Ajax in Action' (online available at: http://www.manning-source.com/books/crane/crane_ch09.pdf) it takes around 16 pages to explain the implementation of this simple component!

ZK solution

Declaration:

```
<?component name="doubleCombo" extends="div"  
    class="com.processwide.demo.zk.components.DoubleComboComponent"?>
```

Usage:

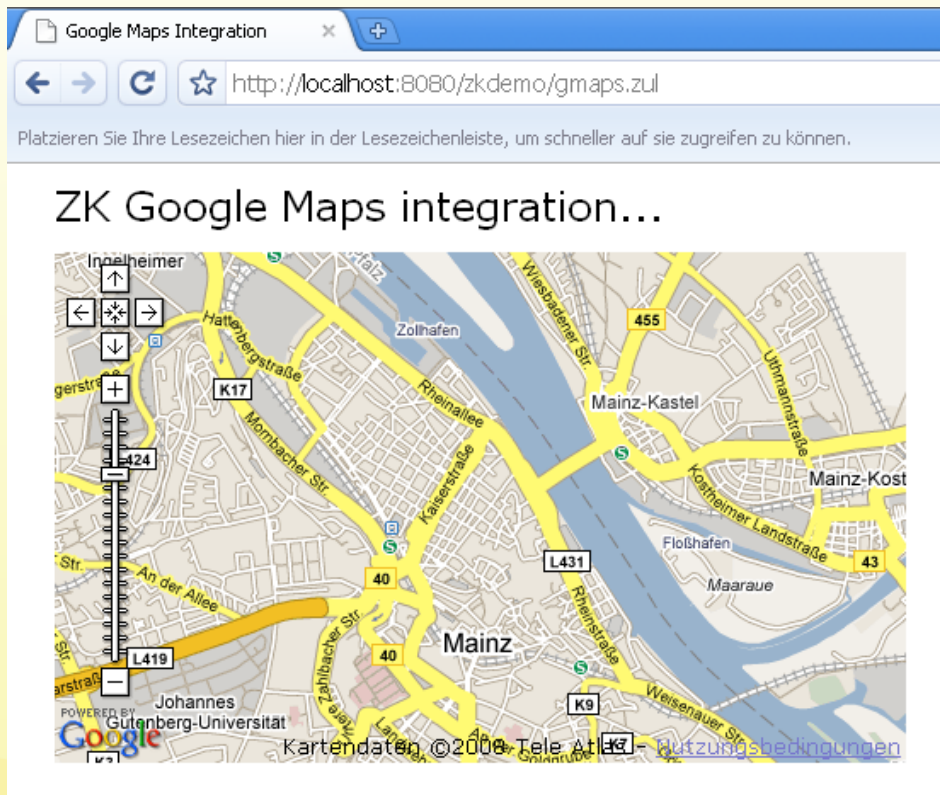
```
<doubleCombo  
    orientation="horizontal"  
    rows="5,5"  
    dataSourceHandler="${componentsWindow.dataSourceHandler}"  
    titleCombo1="Kantone in der Schweiz"  
    titleCombo2="Gemeinden"  
/>
```

Java:

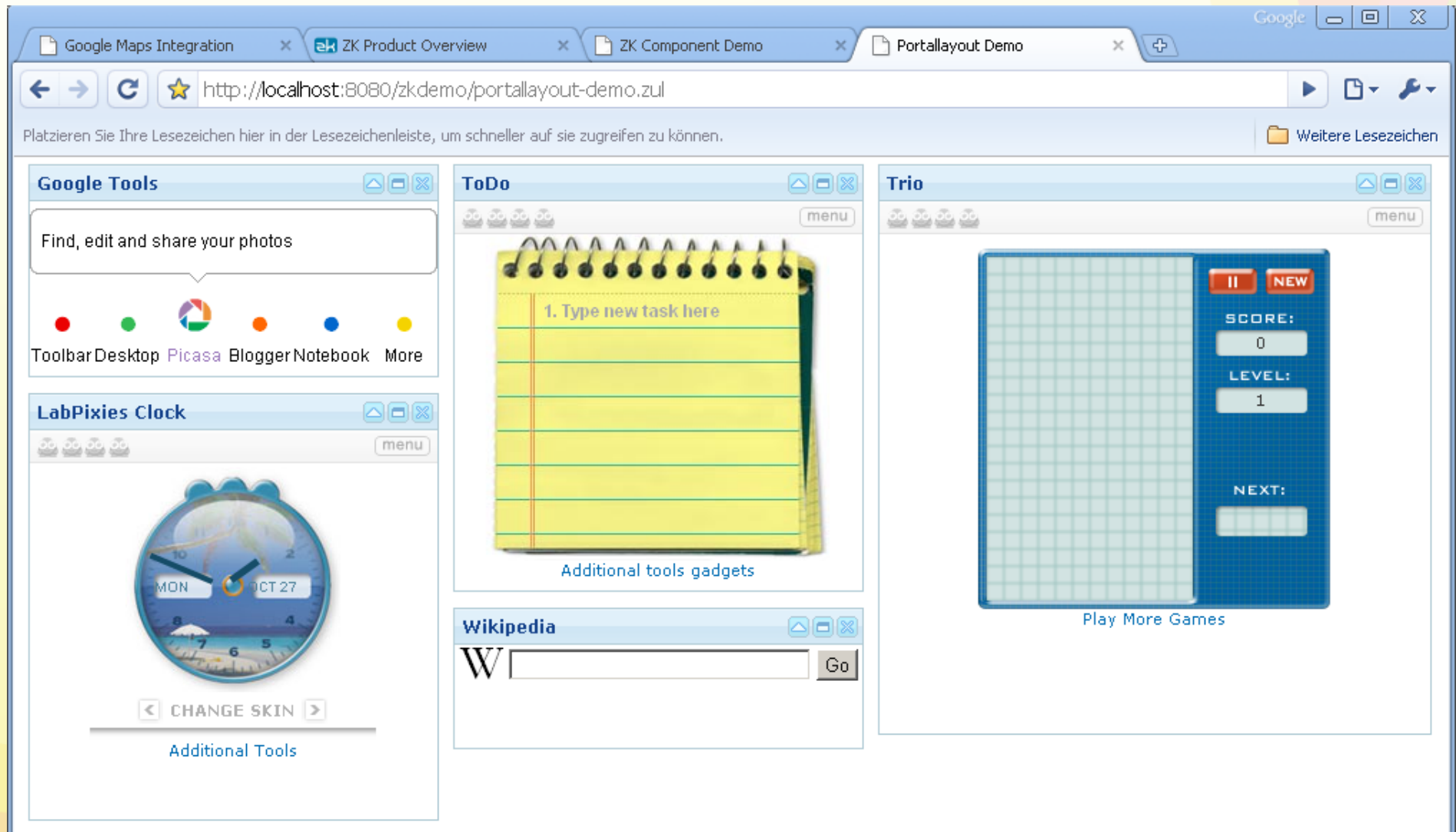
```
package com.processwide.demo.zk.components;  
  
public class DoubleComboComponent extends Div {  
    ...  
}
```

Integration of Gmaps

```
<gmaps width="500px" height="300px" lat="50.00385"
  lng="8.26778" showLargeCtrl="true"/>
```



Demo of Portallayout



Summary

Compared to other AJAX Frameworks ZK is simply outstanding in ...

- Productivity
- Component library
- Building custom components
- Support and development activity of the provider
- Documentation
- Integration with other technologies (JSF, JSP, RSS, GMaps, Dojo, FCKEditor, Timeline, PayPal Service, Portlets, JFreeChart, JasperReports, ...)

Questions

