



Building Rich Web Applications using jMaki

Doris Chen Ph.D.
Staff Engineer/Technology Evangelist
Sun Microsystems, Inc.



Agenda

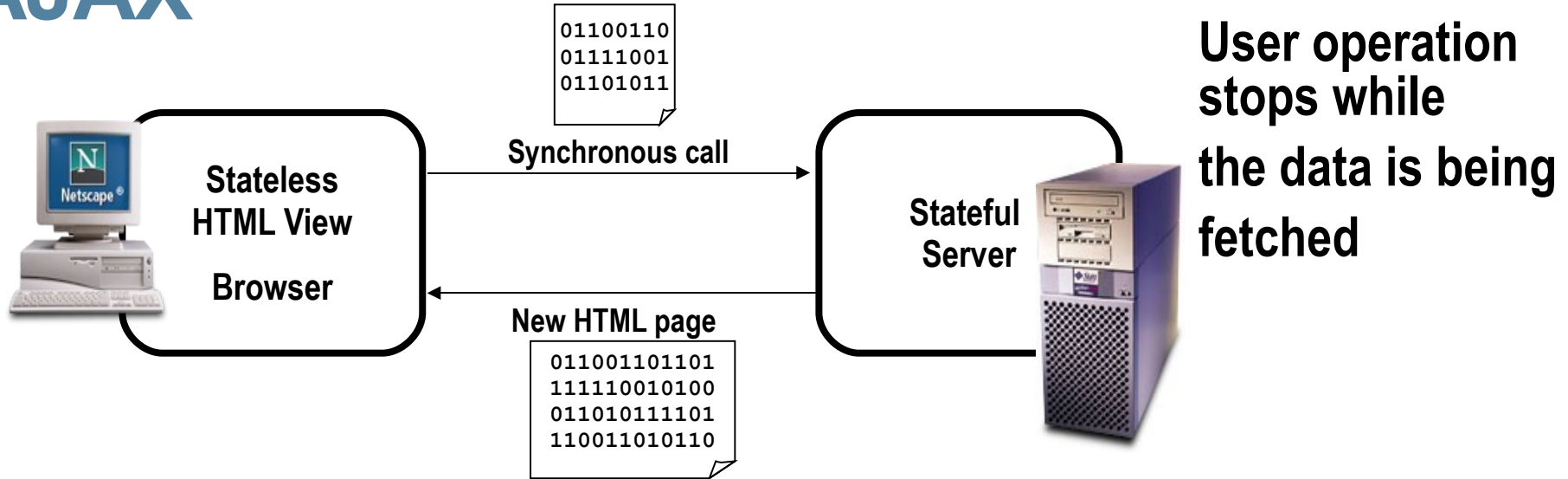
- What is Ajax
- Introduction to jMaki
- jMaki Architecture
- jMaki Recipe
- jMaki Work with External Resources
- jMaki Widgets Work Together
- jMaki Performance Enhancement
- Summary and Resources

What is AJAX?

What is AJAX?

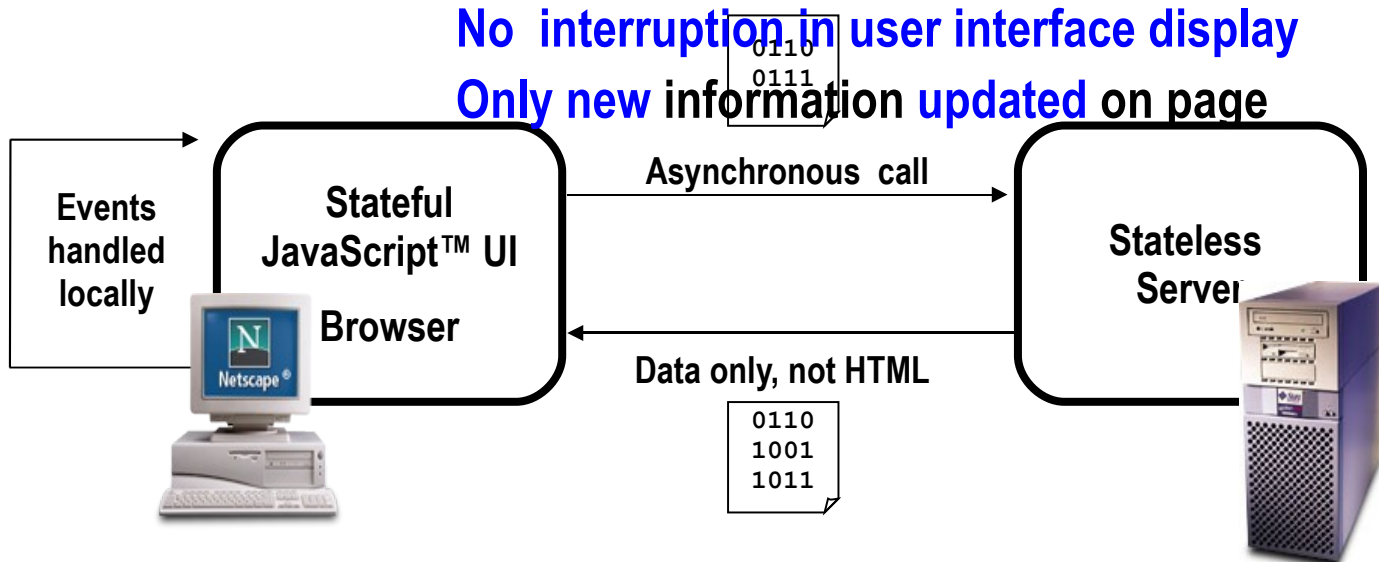
- AJAX= acronym for:
 - > **A**synchronous **J**avaScript and **X**ML
- Browser client uses **JavaScript** to **Asynchronously** get **XML data** from a server
- Now ~ DHTML

User Interface: Traditional Web vs. AJAX

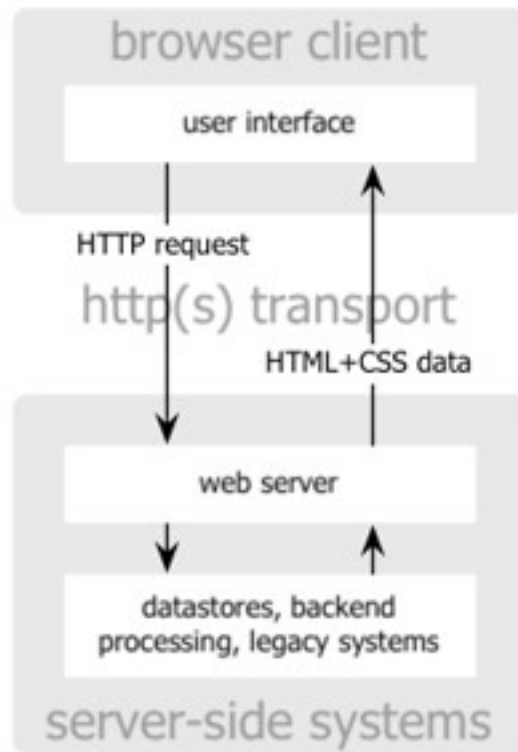


AJAX

No interruption in user interface display
Only new information updated on page



Traditional Web



classic
web application model

AJAX



within a browser,
there is AJAX engine

Ajax
web application model

Current Issues with AJAX

- “Naked” Ajax is too complex
 - > You need a deep understanding of Ajax techniques
 - > Have to handle all XMLHttpRequest interactions yourself
- JavaScript
 - > **inconsistent** support among browsers
 - > requires cross browser testing
 - > code is visible to a **hacker**
 - > Can be **difficult to develop**, debug, and maintain

Solutions

- JavaScript Toolkits
 - > Wrap up ajax details in javascript libraries
 - > jQuery, Dojo, prototype+scriptaculous, Yahoo,...
- Frameworks
 - > DWR
 - > GWT, Wicket
 - > jMaki

Introduction to jMaki

Origins of jMaki?



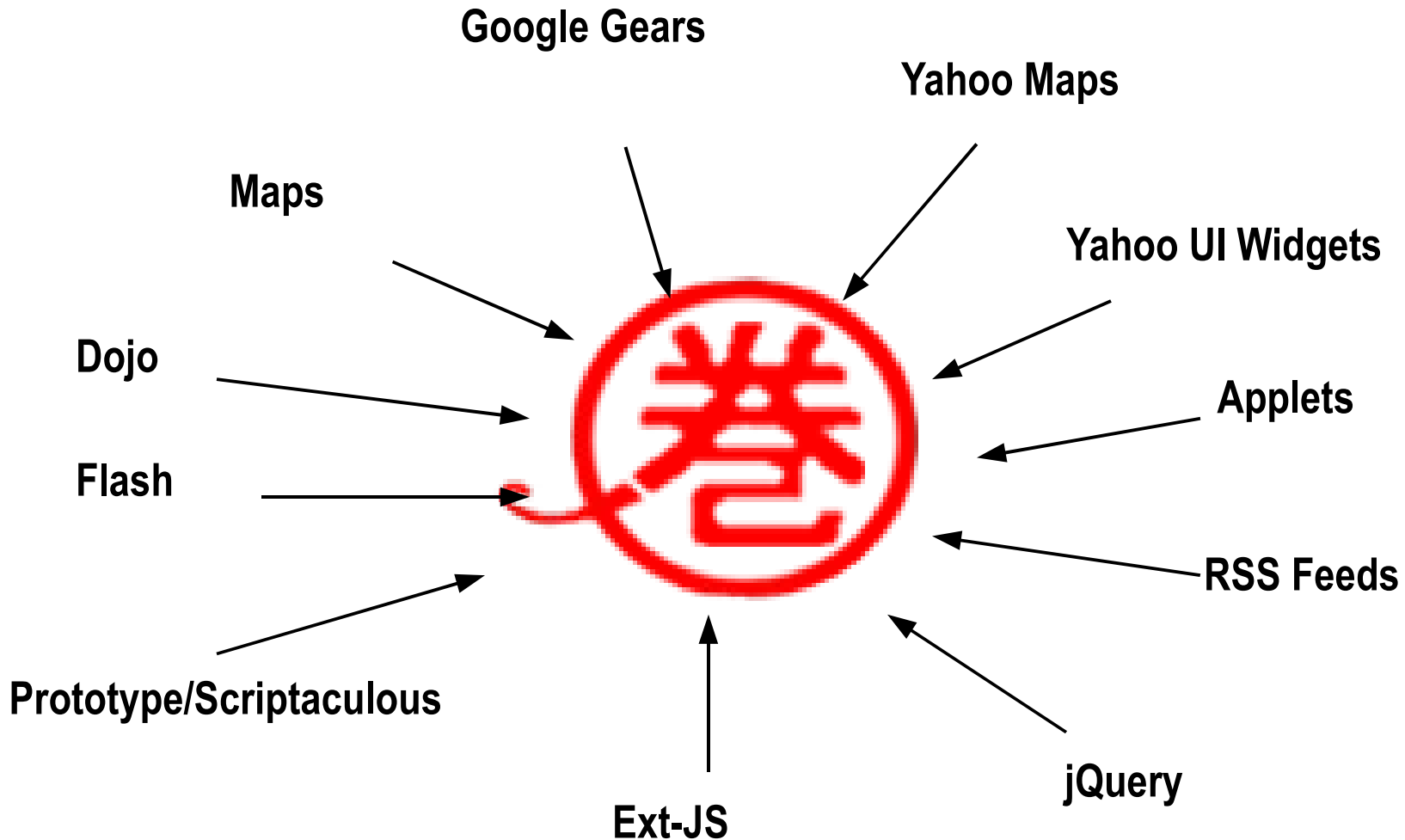
- 'j' is for JavaScript™ technology
- Maki == to wrap in Japanese
- Started as a way of wrapping JavaScript technology functionality
- Project jMaki has evolved to provide more

What Is jMaki

A Client-Server Lightweight Framework for Providing JavaScript Technology-Centric User Interfaces

- Open Source (BSD)
- Allows you to integrate the best of breed JavaScript technology into your applications
- Ajax Application model built in
- A widget model for creating widgets or wrapping existing functionality
- Netbeans™, Eclipse, Ant support

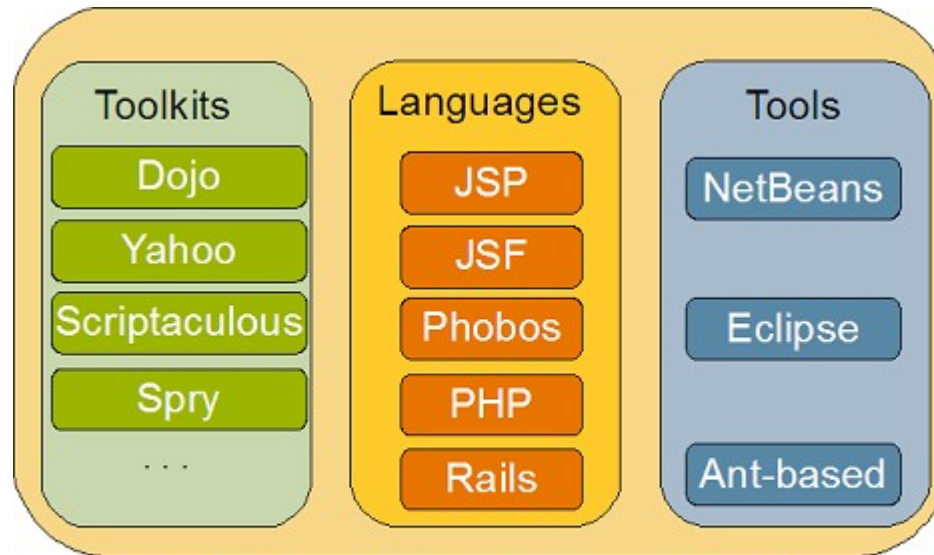
Integrating the Best of Breed Web Assets



Why Use jMaki?

- Defaults set out of the box
 - > Convention over configuration
- Consistent programming model
 - > Universal tag library for widgets from popular toolkits
 - > Standardizes Event / Data Models
 - > Mashup Capabilities built in
- Portable Widget libraries
 - > Share your extensions / widgets libraries
- Ajax Acceleration built in
- Multi-server runtime support
 - > PHP / JSP / JSF / Ruby / Phobos

Benefits of using jMaki



jMaki – Web 2.0 Apps Your Way
(ajax.dev.java.net)

Multi-Server Support

JSP technology: index.jsp

```
<%@ taglib prefix="a" uri="http://jmaki/v1.0/jsp" %>  
<a:widget name="hello" args="{name: 'World'}" />
```

PHP: index.php

```
<?php require_once 'Jmaki.php'; ?>  
<?php addWidget( array( 'name' => "hello",  
                        'args' => "{name: 'World'}" )); ?>
```

Ruby: index.html.erb

```
<%= jmaki_widget 'hello',  
      :args => { :name => name },  
      :value => { 'World' }  
%>
```

Who is using jMaki

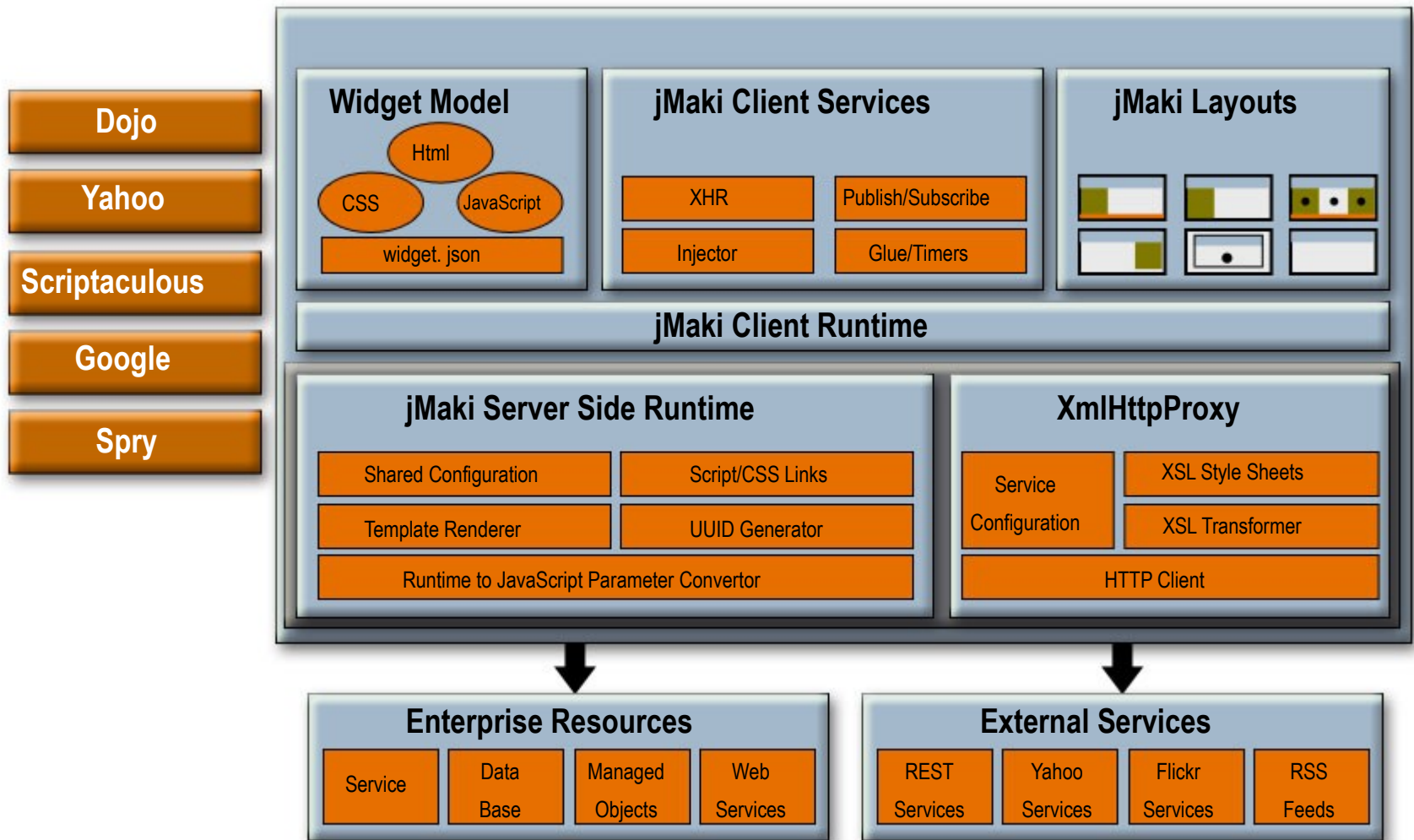
- <http://thespanishsite.com/>
- <http://travelmuse.com>
- <http://jmaki.com>
- <http://zembly.com>
- <http://www.snaioreto.it/home/home.jsp>
- <https://edu-gelc.dev.java.net>

jMaki Architecture

jMaki Framework

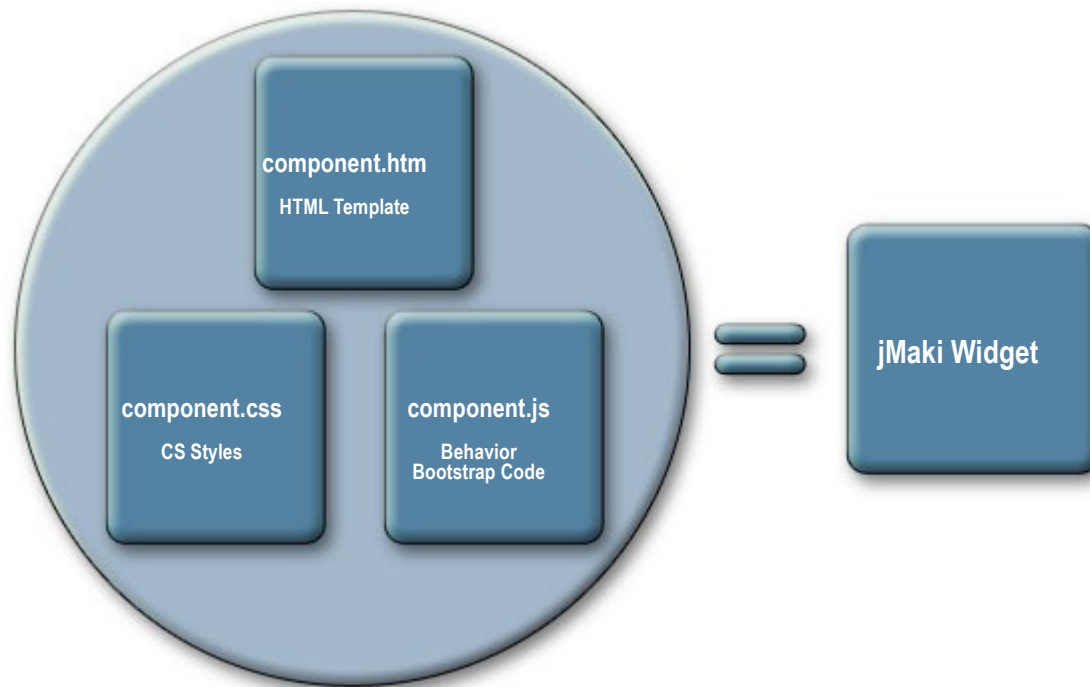
- Widget component / library models
- Data Models
- Events using Publish / Subscribe
 - > Actions / Glue
- Layouts / Theming model
- XmlHttpProxy
- Extensions

jMaki Architecture



Widget Model

Simple



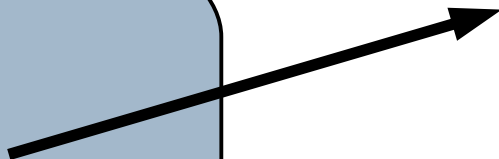
Hello World Widget

component.htm

```
<div id="${uuid}"></div>
```

component.js

```
jmaki.namespace("jmaki.widgets.hello");  
jmaki.widgets.hello.Widget = function(wargs) {  
  var mydiv = document.getElementById(wargs.uuid);  
  mydiv.innerHTML = "Hello " + wargs.args.name;  
}
```



```
id  
value  
service  
args  
selected
```

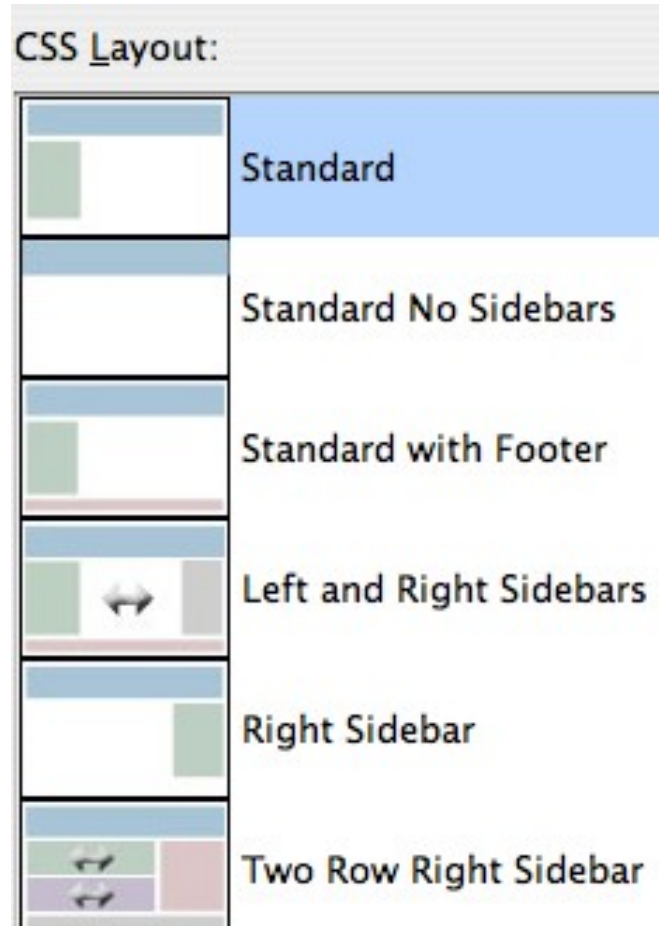
index.jsp

```
<%@ taglib prefix="a" uri="http://java.sun.com/jmaki" %>  
<a:ajax name="hello" args="{name: 'world'}" />
```

jMaki Layouts and Themes

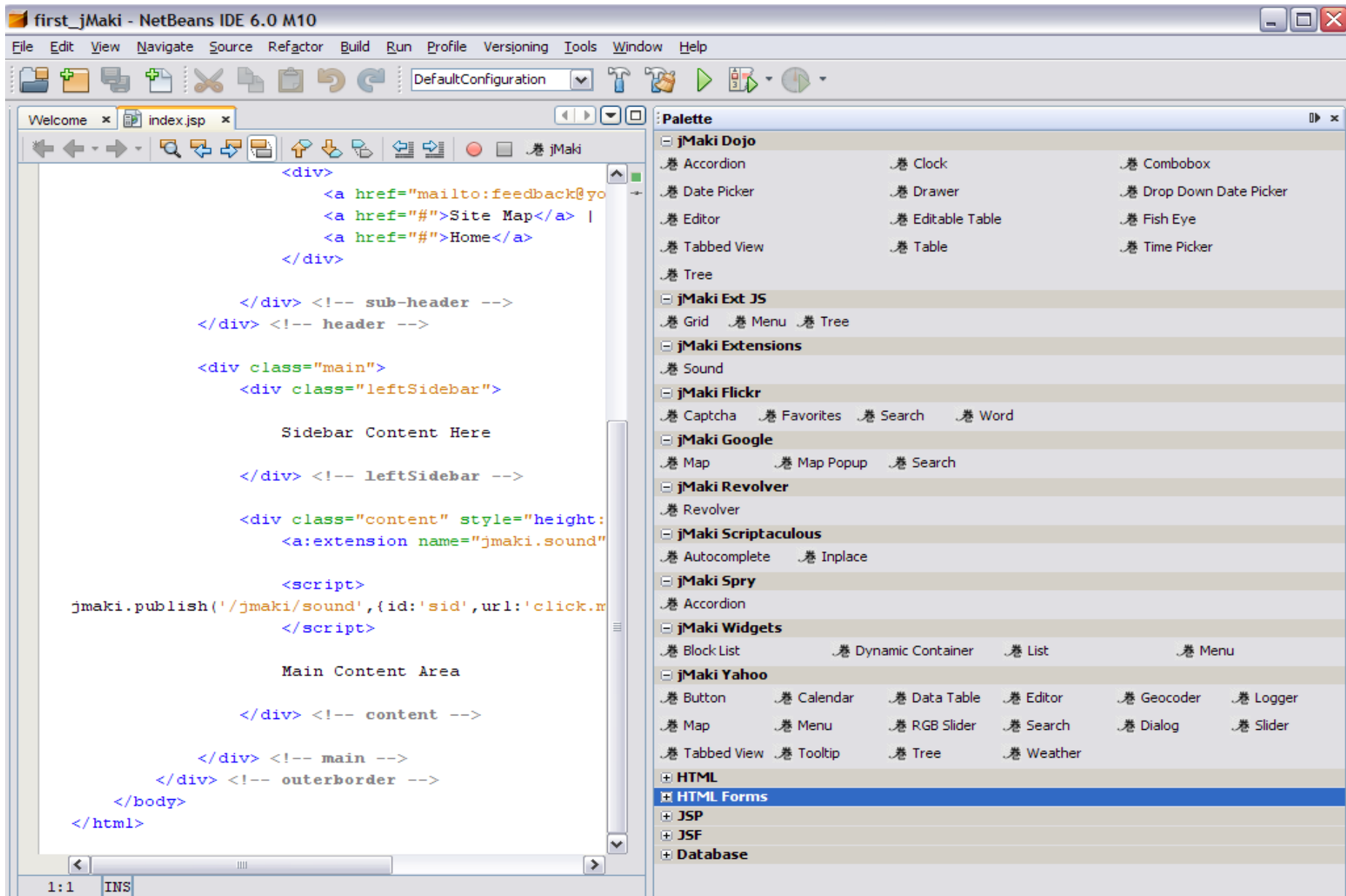
- Layouts
 - > Layouts use common naming conventions
 - > Widgets size to fit the layouts
 - > HTML templates provided for the layouts
- Themes
 - > CSS-based and separate from the layouts
 - > Widgets provide default themes which are overridden

jMaki Layouts



jMaki Recipe

jMaki Palette in NetBeans IDE



The screenshot shows the NetBeans IDE interface with the following components:

- Window Title:** first_jMaki - NetBeans IDE 6.0 M10
- Menu Bar:** File, Edit, View, Navigate, Source, Refactor, Build, Run, Profile, Versioning, Tools, Window, Help
- Toolbar:** Includes icons for file operations, configuration, and execution.
- Editor:** Displays the content of `index.jsp`. The code includes HTML structure with a sidebar and a main content area, and a JavaScript snippet for publishing a sound effect.
- Palette:** A sidebar on the right containing a categorized list of jMaki widgets and extensions. The categories include:
 - jMaki Dojo:** Accordion, Clock, Combobox, Date Picker, Drawer, Drop Down Date Picker, Editor, Editable Table, Fish Eye, Tabbed View, Table, Time Picker, Tree
 - jMaki Ext JS:** Grid, Menu, Tree
 - jMaki Extensions:** Sound
 - jMaki Flickr:** Captcha, Favorites, Search, Word
 - jMaki Google:** Map, Map Popup, Search
 - jMaki Revolver:** Revolver
 - jMaki Scriptaculous:** Autocomplete, Inplace
 - jMaki Spry:** Accordion
 - jMaki Widgets:** Block List, Dynamic Container, List, Menu
 - jMaki Yahoo:** Button, Calendar, Data Table, Editor, Geocoder, Logger, Map, Menu, RGB Slider, Search, Dialog, Slider, Tabbed View, Tooltip, Tree, Weather
 - HTML Forms** (highlighted)
 - JSP**
 - JSF**
 - Database**

DEMO

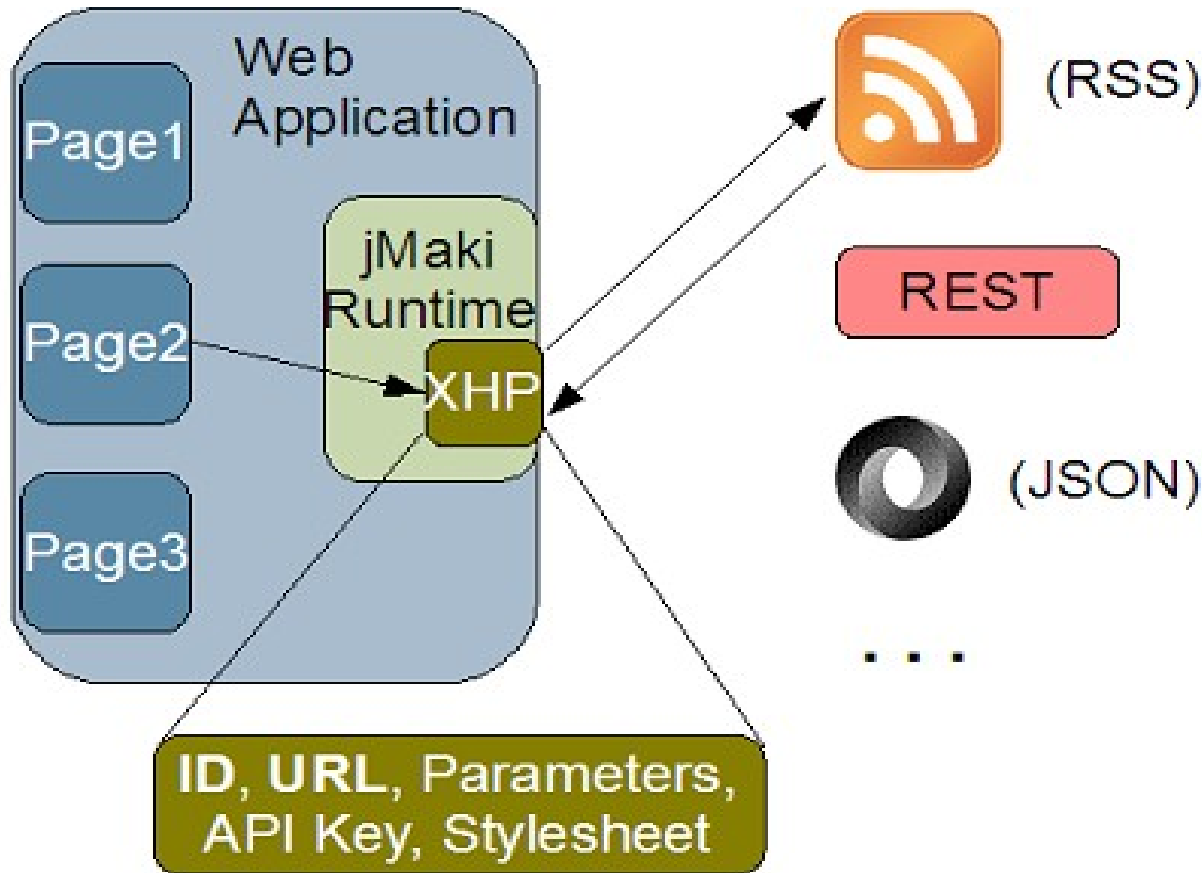
jMaki Widgets

jMaki Recipe

- Choose a layout
- Drop widgets into a layout
- Configure widgets (if necessary)
- Provide glue if widgets interact
- Choose a theme

jMaki Work with External Resources

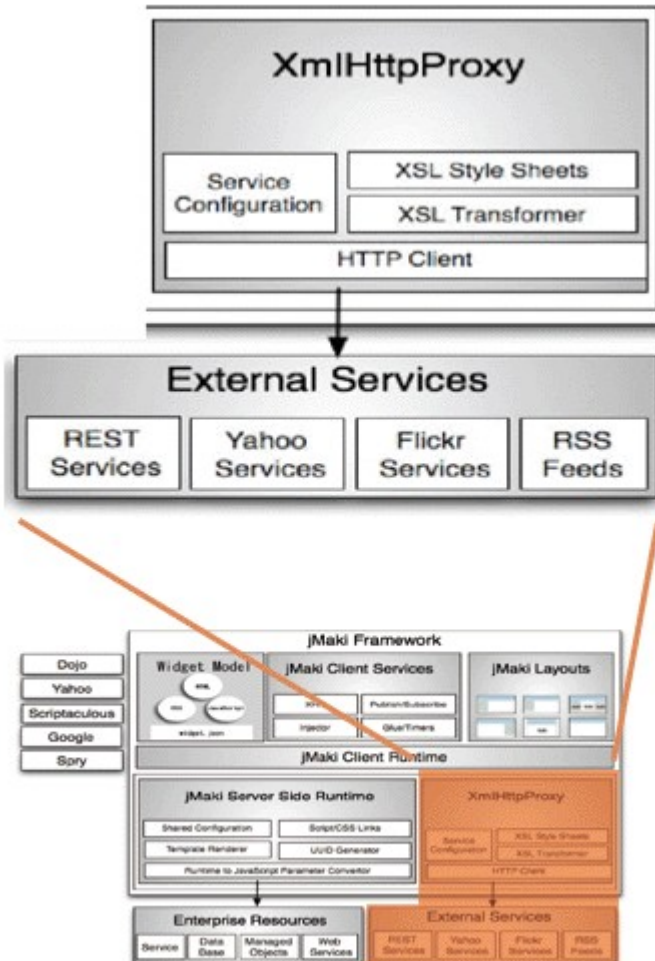
Access External Services



jMaki – Accessing External Services

XMLHttpProxy (XHP)

A Window to the Outside World



- Access to RESTful web services not in the web app domain
 - > Yahoo GeoCoder
- Provides customizable XSL-to-JSON transformations
- Access to RSS feeds
 - > Atom/RSS
- *Widgets are tuned to use it*

Accessing External Services

```
<a:widget name="jmaki.blockList" service="/xhp?id=rss" />
```

"**rss**" configured in the configuration file "**xhp.json**"

xhp.json:

```
{ "id": "rss",  
  "url": "http://weblogs.java.net/blog/ludo/index.rdf",  
  "xslStyleSheet": "rss.xsl"  
}
```

Access Enterprise Resources

jMaki work with JPA

```
<a:widget name="yahoo.dataTable"  
  service="data.jsp" />
```

data.jsp provides dynamic data from persistence using JPA

Access Enterprise Resources

data.jsp dynamic data using JPA

```
<%@ page import="java.util.*" %>
<%@ page import="server.Company" %>
<%@ page import="javax.persistence.*" %>
<%
    EntityManagerFactory emf = Persistence.createEntityManagerFactory("jmaki-jpaPU");
    EntityManager em = emf.createEntityManager();
    List<Company> list = em.createQuery("select c from Company c").getResultList();
    out.println("{columns : [" + "{ label : 'Company Name', id : 'companyName'}," + ... "],");
    out.println("rows: [");
    for (int i=0; i<list.size(); i++) {
        Company c = list.get(i);
        out.print("{ companyName: " + c.getCompanyname() + "," + "price: " + c.getPrice() + "," + "change: " +
c.getChange() + "," + "pctChange: " + c.getPercentchange() + "," + "lastUpdated: " + c.getLastupdated() + "}");
        if (i < list.size()-1)
            out.println(",");
        else
            out.println();
    }
    out.println("] }");
%>
```

DEMO

Work with external resources

jMaki Widgets Talk to Each Other

How the widgets talk to each other?

jMaki Events

- **Publish/Subscribe** is much like a server-side messaging system by it runs in the scope of an HTML page
- **topics**: a means for multiple jMaki widgets to communicate with each other in a page
- **Action**: declarative events
 - > Action property
 - > No additional JavaScript code
- **Glue**: programmatic events
 - > Allows you to provide unobtrusive application behavior in a single place
 - > Ties the widgets together (loosely) based on topic names

Action Example

```
<a:widget name="dojo.fisheye" value="[  
{ iconSrc:'https://ajax.dev.java.net/images/blog_murray.jpg',  
  label : 'You are here!',  
  action : { topic : '/foo/select', message : {targetId : 'bar'}}  
},  
{ iconSrc:'https://ajax.dev.java.net/images/chinnici.jpg', label : 'test3'}  
]"/>
```

```
<a:widget name="dojo.tabbedview"  
  subscribe="/foo"  
  value="{items:[  
    {label : 'My Tab', content : 'Some Content'},  
    {id : 'bar', label : 'My Tab 2', include : 'test.jsp ', lazyLoad : true },  
    {label : 'My Tab 3', content : 'More Content', selected : true}  
  ]}" />
```

jMaki Glue: Publish/Subscribe

- jMaki widgets communicate within the page in JavaScript programming language via jMaki **glue**
- Glue:
 - > JavaScript technology-based and loaded application-wide or based on a page
 - > JavaScript technology handlers mapped to topic names
- Steps to using the **glue** mechanism
 - > Declare the topic you want to subscribe or listen to
 - > Declare the name of the function (listener) which will handle the notification
 - > Provide the code to handle the notification
- Widgets configured to work by default

Publish/Subscribe

- A means for jMaki widgets to communicate with each other in a page using topics
 - > Much like server-side messaging system but runs in the scope of HTML page

Publish (index.jsp)

```
<a:widget name="yahoo.button"
publish="/button/addTable"
args="{label : 'Add table data' }"/>
```

Subscribe (glue.js)

```
jmaki.subscribe ("/button/addTable/onClick",
"jmaki.listeners.addTableData");
```

```
jmaki.listeners.addTableData = function(args) {
//set the row data
var row = [
{ title : 'Book Title 3',
author : 'Author 3',
isbn: '4415',
description : 'A Some long description'
}, ... ];
...
}
```

DEMO

Widgets working together

jMaki Webtop

jMaki Webtop

- Problem Trying to Solve
 - > Create Mashups
- Features Needed
 - > Create a Ajax User interface
 - > Manage Widgets, Widget State
 - > Manage Users
 - > Integrate Services

jMaki Webtop

- View
 - > JavaServer Pages
 - > JSP tag based jMaki
- Widget Management
 - > Java Persistence: MySQL(server) Google Gear (client)
- Key feature
 - > Simple & easy to use (drag and drop, event/action configure at Browser level)
 - > Extensible (add your own widgets & gadgets in Palette)
 - > Widget Customization (property, position, customize own widgets)
 - > Shared (import/export)

jMaki Webtop (Brazilian Soccer)

Browser window: jMaki Webtop
URL: http://localhost:8080/Soccer_Player_Mashup/jmaki-webtop-soccer/test/index_1.html#

Getting Started | Latest Headlines

卷 jMaki Webtop - Welcome

Add Widgets | Webtops | Options | jMaki.com | About

Flow

Pele

Google Map

Satellite | Map | Hybrid

©2008 TerraMetrics, Map data ©2008 MapLink/Tele Atlas - Terms of Use

Flash Player

Tag Cloud

Rivaldo Pele Ronaldo Didi Garrincha Romario Ronaldinho Zico

Done

jMaki WebTop Mashup (Wedding

The screenshot shows a web browser window titled "jMaki Webtop" with the URL http://localhost:8080/WeddingMashUp_0.9/jmaki-webtop/test/index.html. The page features a header with the jMaki logo and the text "jMaki Webtop - Wedding". A "Palette - All Widgets (Drag Me)" bar contains several widgets: Editor, Event Calendar, Flow, Revolver, and Flash Player. The main content area includes a "Google Map" widget showing a location with two red pins, a "Flash Player" widget with a video of a couple in traditional attire, and a "Flow" widget displaying a sequence of three images related to a wedding ceremony. A "Tag Cloud" widget at the bottom lists tags: Kimono, Ceremony, Sakura, Castle, Happy, NightView, and Greg. The browser's status bar at the bottom left shows "Done".

jMaki Performance Enhancement

What problem are we trying to solve?

- Network Performance
 - > Many CSS and JavaScript resources
 - > Increase the number of HTTP requests
 - > Slow down the initial page load
 - > Large CSS and JavaScript require large bandwidth to transfer client
- Page Rendering
 - > Stop or render slower when JavaScript and CSS resources are loaded over a network
- Need solution to improve both the network performance and browser loading of Ajax based applications

jMaki Performance Enhancer

- Available since jMaki 1.8
- Combined Resources
 - > Reduce the number of network calls by **combining** CSS and JavaScript resources
 - > Combine CSS code and place it at the beginning of the page
 - > Combine JavaScript code and place it at the end of the page
- Minified Resources
 - > Automatically load a minified **component-min.js** in place of component.js
 - > Use minified JavaScript if available
- Headers
 - > Set the correct headers to ensure cacheable resources are correctly cached locally by browsers

JSP Page: Add Performance Tag **a:page**

```
<html>
  <head> ...
    <a:page>
  </head>
  <body> ...
    <div id="main">
      <div id="rightColumn" style="height:400px">
        <a:widget name="jmaki.blockList" ... />
      </div> <!-- end rightColumn -->
      <div id="leftColumn" style="height:400px">
        <a:widget name="jmaki.tagcloud" .../>
      </div> <!-- leftColumn -->
    </div> <!-- main -->...
  </a:page></body></html>
```

Edit web.xml

```
<context-param>
```

```
  <param-name>jmaki-combinescripts</param-name>
```

```
  <param-value>>true</param-value>
```

```
</context-param>
```

```
<context-param>
```

```
  <param-name>jmaki-combinestyles</param-name>
```

```
  <param-value>>true</param-value>
```

```
</context-param>
```

```
<servlet>
```

```
  <servlet-name>Combined Resource Servlet</servlet-name>
```

```
  <servlet-class>jmaki.runtime.CombinedResourceServlet</servlet-class>
```

```
  <load-on-startup>2</load-on-startup>
```

```
</servlet>
```

```
<servlet-mapping>
```

```
  <servlet-name>Combined Resource Servlet</servlet-name>
```

```
  <url-pattern>/cr</url-pattern>
```

```
</servlet-mapping>
```

DEMO

jMaki Performance Enhance Demo

Performance Data

- More widgets, more performance gain
- Up to 30% reduce in component upload
- Your mileage will be different

Summary and Resources

Summary

- jMaki allows you to easily create rich web applications
- jMaki allows you to use the best Ajax components and toolkits
- Consistent data and event models
- jMaki supports server side implementation for multiple languages and platforms

Resources

- Main jMaki site
 - > <https://ajax.dev.java.net/>
- jMaki Widgets site
 - > <https://widgets.dev.java.net/>
- jMaki Hands on Lab
 - > <http://developers.sun.com/learning/javaoneonline/j1lab.jsp?lab=LAB-4530&yr=2008&track=1>
- Send your questions to the public mailing lists and forums
 - > jMaki mailing alias <mailto:dev@ajax.dev.java.net>
 - > jMaki Forum <http://forums.java.net/jive/forum.jspa?forumID=96>
- NetBeans IDE
 - > <http://netbeans.org/>
- Project GlassFish
 - > <http://glassfish.dev.java.net/>



Building Rich Web Applications using jMaki

Doris Chen Ph.D.

doris.chen@sun.com

Staff Engineer/Technology Evangelist

Sun Microsystems, Inc.

