AM09

Web 2.0 Development with IBM WebSphere sMash

Matthew Perrins

Executive IT Specialist

Software Group Lab Services

matthew_perrins@uk.ibm.com

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

IBM

Rational. software

What is WebSphere sMash?

WebSphere sMash is an Agile Web Application Platform

- Architected around Dynamic Scripting, REST, Rich Web Interfaces, AJAX, and Feeds
- Optimized for
  - **Speed**
  - **Simplicity**
  - **Agility**

Key Scenarios

- Enables developers to build web 2.0-style applications by easily pulling in, composing, and “cobbling together” pre-existing assets (PHP assets, services, feeds, code snippets) using dynamic scripting languages and simple consumption principles based on REST.
- Leverages existing SOA investments by enabling rapid development of dynamic web applications that are assembled from enterprise assets and publicly available APIs.

WHAT IS PROJECT ZERO?

Project Zero is the development and incubation community for WebSphere sMash

- Live on the Internet since June 2007

Project Zero represents

- The people that build and use WebSphere sMash
- The incubation of new technology that will deliver in future versions of WebSphere sMash
- The community of 3rd party assets that leverage the WebSphere sMash platform

All released versions are called WebSphere sMash

WEBSPHERE SMASH AND THE WEB 2.0 STRATEGY

WebSphere sMash fits perfectly into WebSphere’s overall SOA Web 2.0 strategy

The WebSphere Web 2.0 Strategy

Unleash enterprise content so it is more easily accessible

- WebSphere is REST-enabling its product portfolio, including
  - MQ, Commerce, WSRR, Web 2.0 FP, WPS, WESB, Datapower,

Leverage this content by enabling agile web applications

- Now that the content is unleashed, you can agilely build web applications to leverage that content as well as content from other backend systems, supplier systems or the web.
- WebSphere sMash fits here, along with Lotus Mashups and InfoSphere MashupHub, as an agile platform for application creation and deployment
- WebSphere sMash can also be used here to build and deliver components such as widgets for Lotus Mashups, or feeds for InfoSphere MashupHub

Run, manage and host agile applications

- Application-centric runtimes (like WebSphere sMash) and management systems (like WebSphere XD) will help you cost effectively run and manage these growing number of these web applications.

PACKAGING

The technology is WebSphere sMash is available in a variety of packages

Package Name	Description
WebSphere sMash	Production version of the WebSphere sMash Platform. Standard IBM commercial license. Available through normal IBM channels.
WebSphere sMash Reliable Transport Extensions	Production version of the extended features in sMash related to messaging and reliable communications. Standard IBM commercial license. Available through normal IBM channels.
WebSphere sMash Developer Edition (DE)	This is the community version of the exact same code you get with WebSphere sMash. WebSphere sMash DE represents the shipped and stable version of the product for developers to use to build applications.
Project Zero	This is the community version of the latest and greatest unreleased technology that is not in a WebSphere sMash version yet. This is the bleeding edge incubation of new features.

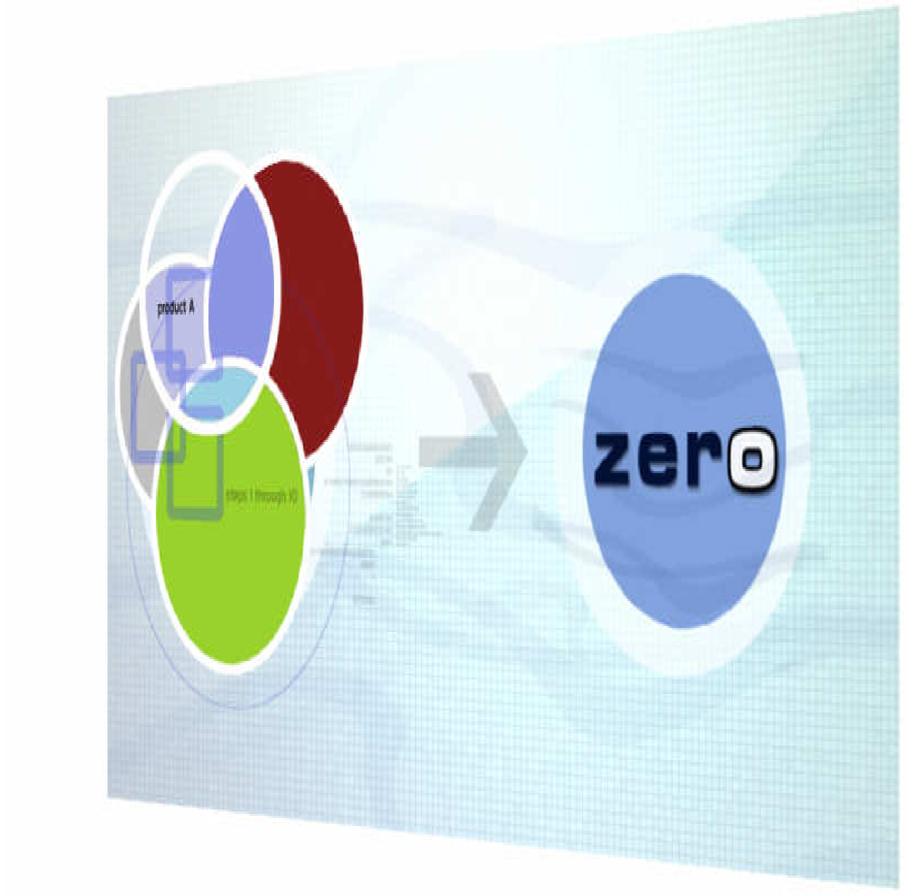
Technology Overview

THE WEBSPHERE SMASH CORE VALUES

Speed

Simplicity

Agility



CORE APPLICATION CONSTRUCTS

EASY FOR DEVELOPERS TO ACCESS, LEARN, AND USE

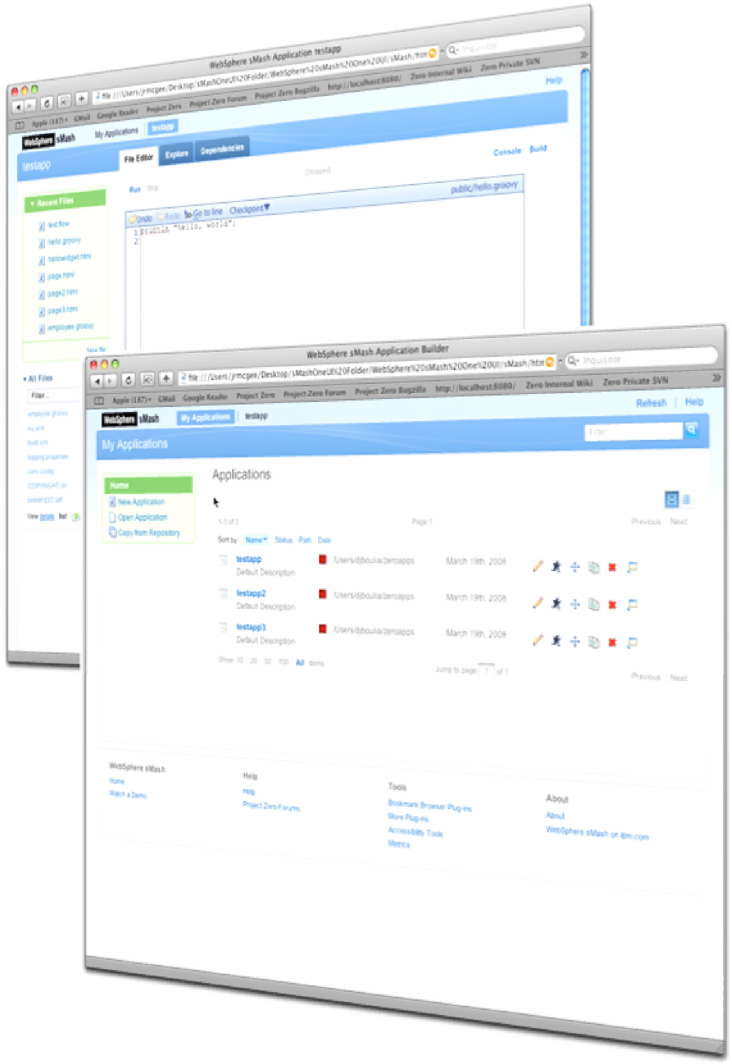
Dynamic Scripting and Templates

Effortless creation of RESTful Services and Data Feeds (RSS, ATOM)

Simple Event-based execution environment

State externalized into a shared memory space (Global Context)

Repository of pre-built Services and Libraries provides useful building blocks



DYNAMIC SCRIPTING

- WebSphere sMash is a dynamic scripting platform
- Application Logic is created in one of two scripting languages
  - Groovy (for people that prefer Java)
  - PHP (for the 3 Million existing PHP programmers)
- Java is positioned as the "system" language
  - Mostly used to implement system extensions and application libraries
  - Entire applications can be written in Java, if desired
    - Requires more configuration

APPLICATION CENTRIC RUNTIME

WebSphere sMash is an application-centric runtime

- You create an application and run it
- You do not package an application and deploy it to a multi-application server
- Each application runs in its own process (JVM)
- Runtime is designed to be short lived
  - Update recycles after idle timeout or max number of requests

WebSphere sMash is a full stack runtime

- Everything needed to run the application is built in
  - Including the HTTP stack
- No external proxy or web server is required
- Does not deploy as a WAR file inside another JEE container
- An external proxy is used for clustering and multi-app routing

PHP SUPPORT

- Gartner predict that within 5 years 60% of the 5.5 million PHP programmers will work in corporate IT. (Up from 13% of 3M today).
- The PHP runtime is built on top of a standard JVM
  - Supports use of many PHP Extensions
    - XAPI-C interface allows C-based extensions
    - XAPI-J interface allows Java based extensions
  - Supports bridging between Java and PHP
  - Currently supports a subset of PHP
    - The goal is maximum re-use of existing PHP scripts
- PHP runtime provided directly by WebSphere sMash, not php.net
- The goal of PHP support is about unleashing the 3 Million PHP programmers together with the vast library of existing PHP script code and extensions and bringing it to the sMash programming model
- A number of popular PHP application now run on the sMash PHP runtime, including
  - **phpBB**
  - **SugarCRM**

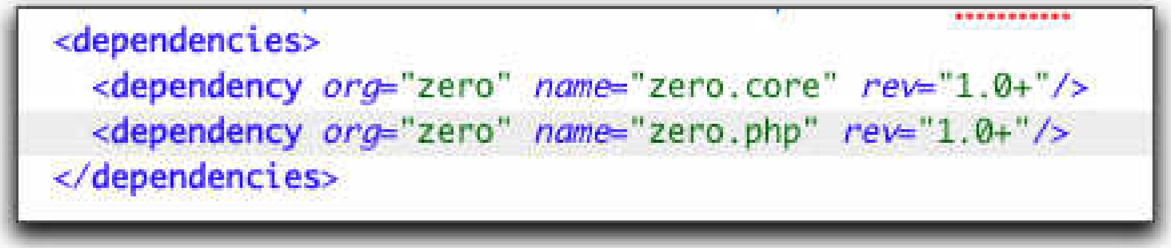
MODULAR ARCHITECTURE

WebSphere sMash applications are based on a very small core

- 5.4 MBytes (includes Groovy).
  - PHP adds additional 14.5 Mbytes
    - Contains 3 platforms currently
- Core provides all of the framework and runtime support, including HTTP transport

Additional features provided in downloadable modules

- Applications declare a dependency on desired features (using Ivy)
- A package management system manages your dependencies, including:
  - The ability to share dependencies on a machine
  - The ability to demand load missing dependencies from the network
  - The ability to manage updates to dependencies that you are using



```
<dependencies>
  <dependency org="zero" name="zero.core" rev="1.0+"/>
  <dependency org="zero" name="zero.php" rev="1.0+"/>
</dependencies>
```

SIMPLE DEPLOYMENT

Essentially the deployment is Zip and Copy

No machine specific information bound into the application

Default mode is shared dependencies

- Application dependencies are load for the deployment machines local repository and pulled off the network if needed

Standalone mode is supported as well

- All application dependencies are included in the ZIP and nothing is needed on the target machine except a JVM

Provides a packaging command to simplify the creation of the ZIP file for deployment

- zero package for shared mode
- zero package standalone for standalone mode

RUNTIME CHARACTERISTICS

Instant On

- Application Available for Service in less than 1 sec
  - 0.672 seconds on a MacBook Pro
- Application JVM starts in about 1 second
  - 1.3 seconds on a MacBook Pro

Clean

- Graceful recovery, isolation, tolerates “bad” code
- Short lived processes
  - Runs for fixed number of requests or idle timeout then restarts
  - No state lost on restart

Cheap

- Cost effective to run in small and large quantities
- Idle Application Footprint ~380 Kbytes
- Running Application JVM ~28 Mbytes TODAY

Supported on “stock” JVM

- IBM, Sun, Mac, etc - Any JSE 5 or 6 JVM

BUILT-IN DEVELOPMENT TOOLING

WEBSPERE SMASH LETS DEVELOPERS BUILD APPLICATION DIRECTLY ON THE WEB

Browser-Based Development IDE

Built as a sMash application

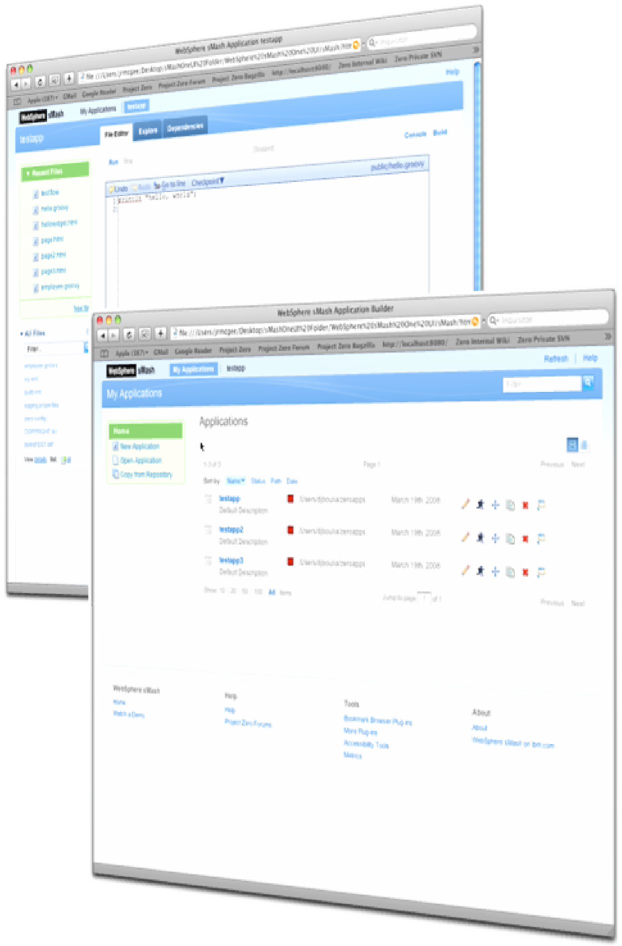
Provides full development lifecycle for Zero applications

- Create, Edit, Test

Provides Visual Editors for Activities and Web Page construction

- Including a DOJO-enabled page editor

Basic Eclipse-based tooling also available if required



WEBSPHERE SMASH ACTIVITIES
LETS DEVELOPERS VISUALLY “MASH-UP” SERVICES AND FEEDS

Assemble-style Development

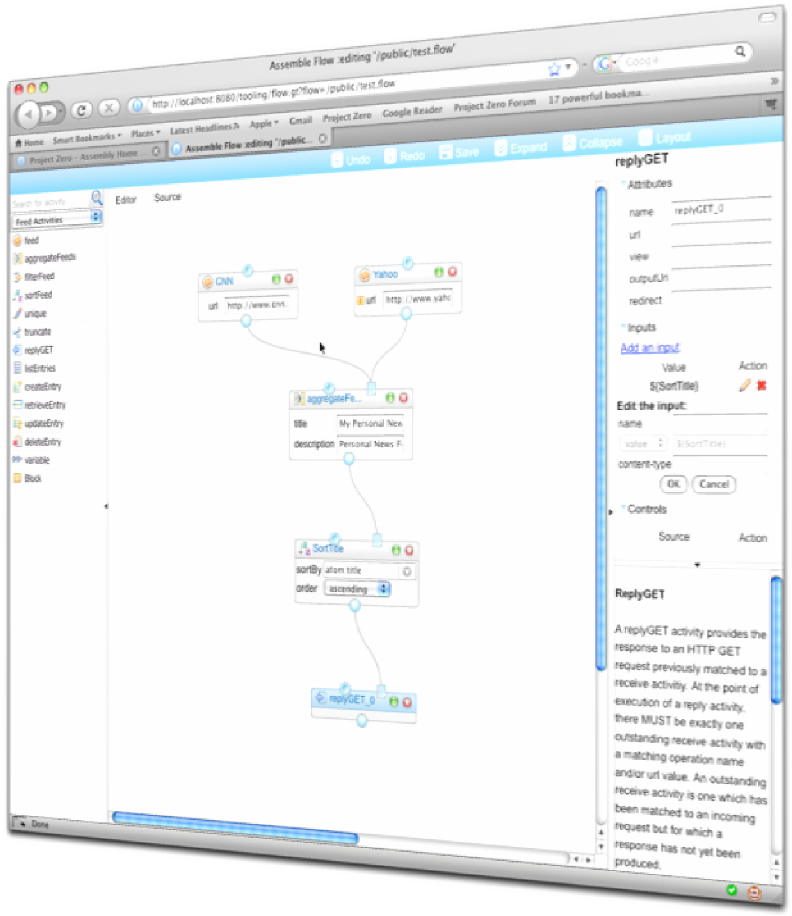
Compose applications by “wiring” together REST services

Visually or programmatically combine existing feeds and services that enrich, sort, and filter data in a pipeline

Configure templates to alter pipeline routes, log events along the pipeline

Numerous built-in activities, including

Get Feed, Call Service, Aggregate, Sort, Transform, Filter, Send Mail, XSLT, Conditionals, Loops



name replyGET_0

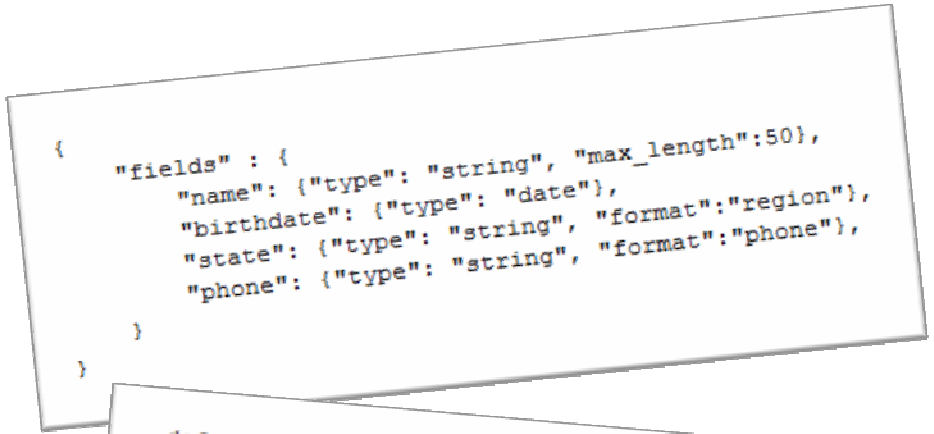
Value $(SortTitle)

ReplyGET

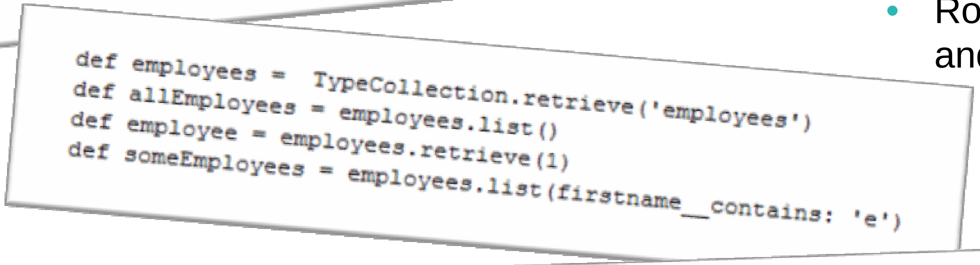
A replyGET activity provides the response to an HTTP GET request previously matched to a receive activity. At the point of execution of a reply activity, there MUST be exactly one outstanding receive activity with a matching operation name and/or url value. An outstanding receive activity is one which has been matched to an incoming request but for which a response has not yet been produced.

RAPIDLY EXPOSE DATA RESTFULLY

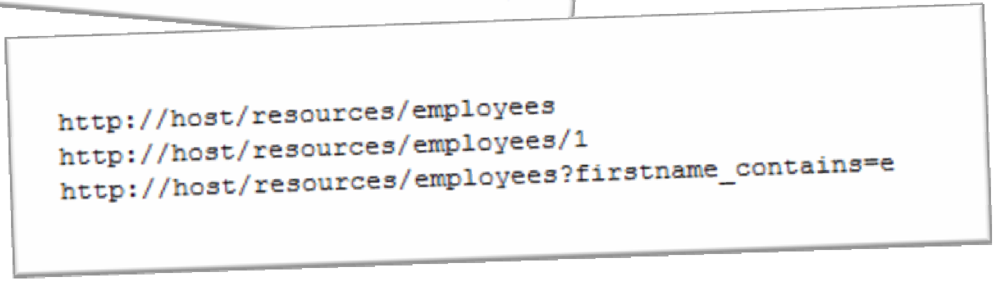
ZERO RESOURCE MODEL ENABLES DEVELOPERS WITH A SIMPLE PROGRAMMATIC AND HTTP DATA API



```
{
    "fields" : {
        "name": {"type": "string", "max_length":50},
        "birthdate": {"type": "date"},
        "state": {"type": "string", "format":"region"},
        "phone": {"type": "string", "format":"phone"},
    }
}
```



```
def employees =  TypeCollection.retrieve('employees')
def allEmployees = employees.list()
def employee = employees.retrieve(1)
def someEmployees = employees.list(firstname__contains: 'e')
```



```
http://host/resources/employees
http://host/resources/employees/1
http://host/resources/employees?firstname_contains=e
```

Model application data

- Constrained set of APIs encourage a RESTful application architecture
- Data model that maps well into Atom feeds and JSON formats
- Robust framework for persistence, validation, and serialization

AVAILABLE MODULES

- There are approximately 65 modules available currently
- Modules provide function in many categories
  - Data Formats (JSON, ATOM, RSS, XML)
  - Data Access
  - Resource Modeling
  - Security / Content Filtering
  - Activity Flows
  - Services
    - Amazon ECS, Flickr, Weather, etc
  - Utilities (such as HTML parsing)
  - Management Tools
  - Development Tooling
  - Reliable Transport Engine for Messaging Interactions

COMMUNITY DRIVEN COMMERCIAL DEVELOPMENT

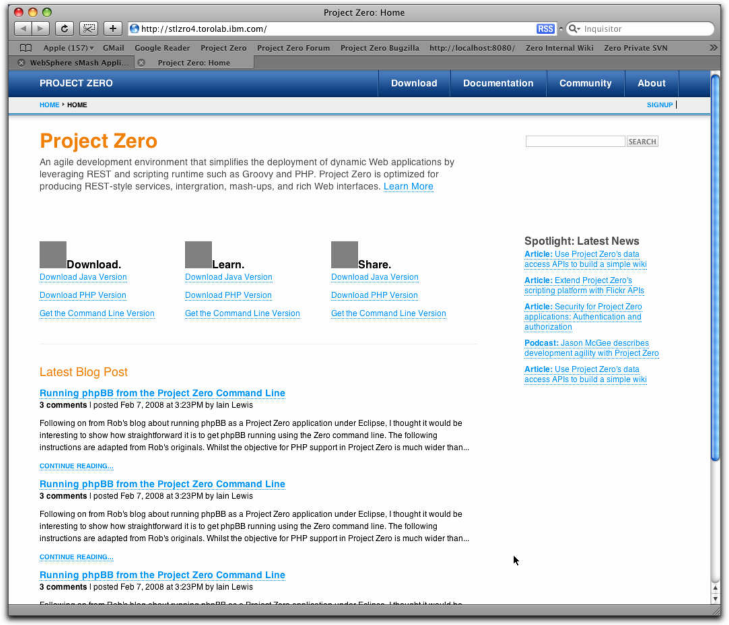
Evolve the core platform based on developer feedback

Commercial development using a transparent development process

Enabled via an external web site providing:

- *A focal point for all sMash development activities*
  - Expose the IBM development process to the external developer community
  - All design decisions are discussed and communicated via external forums
  - Registered users can post comments and feedback to the forums
- *Frictionless download of latest code and documentation*
  - Registration not required for binary downloads
  - Latest builds immediately available to developers
  - Source code can be viewed by registered users

http://www.projectzero.org


Project Zero: Home
http://stlzro4.torolab.ibm.com/
RSS
Inquisitor
Apple (157) GMail Google Reader Project Zero Project Zero Forum Project Zero Bugzilla http://localhost:8080/ Zero Internal Wiki Zero Private SVN
WebSphere sMash Appli...
Project Zero: Home
PROJECT ZERO
Download
Documentation
Community
About
HOME › HOME
SIGNUP
SEARCH

Project Zero

An agile development environment that simplifies the deployment of dynamic Web applications by leveraging REST and scripting runtime such as Groovy and PHP. Project Zero is optimized for producing REST-style services, intergration, mash-ups, and rich Web interfaces. Learn More

Download.	Learn.	Share.
Download Java Version	Download Java Version	Download Java Version
Download PHP Version	Download PHP Version	Download PHP Version
Get the Command Line Version	Get the Command Line Version	Get the Command Line Version

Spotlight: Latest News

Article: Use Project Zero's data access APIs to build a simple wiki

Article: Extend Project Zero's scripting platform with Flickr APIs

Article: Security for Project Zero applications: Authentication and authorization

Podcast: Jason McGee describes development agility with Project Zero

Article: Use Project Zero's data access APIs to build a simple wiki

Latest Blog Post

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

Following on from Rob's blog about running phpBB as a Project Zero application under Eclipse, I thought it would be interesting to show how straightforward it is to get phpBB running using the Zero command line. The following instructions are adapted from Rob's originals. Whilst the objective for PHP support in Project Zero is much wider than...

CONTINUE READING...

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

Following on from Rob's blog about running phpBB as a Project Zero application under Eclipse, I thought it would be interesting to show how straightforward it is to get phpBB running using the Zero command line. The following instructions are adapted from Rob's originals. Whilst the objective for PHP support in Project Zero is much wider than...

CONTINUE READING...

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

PROJECTZERO.ORG
LOWERING THE BARRIERS TO COMMUNITY ACCESS

Anonymous Visitors can…

Browse the site
View Wiki content
Read Forums
Search the Bug Database
Read Blogs
Download Binary Drivers*

Focused on easy access

- Internet web site
- Free access to the platform

Registered Users can…

Post to the Forum
Submit Bug Reports
Submit Feature Requests
Comment on Blog Posts
Access Source Code*

Focused on feedback

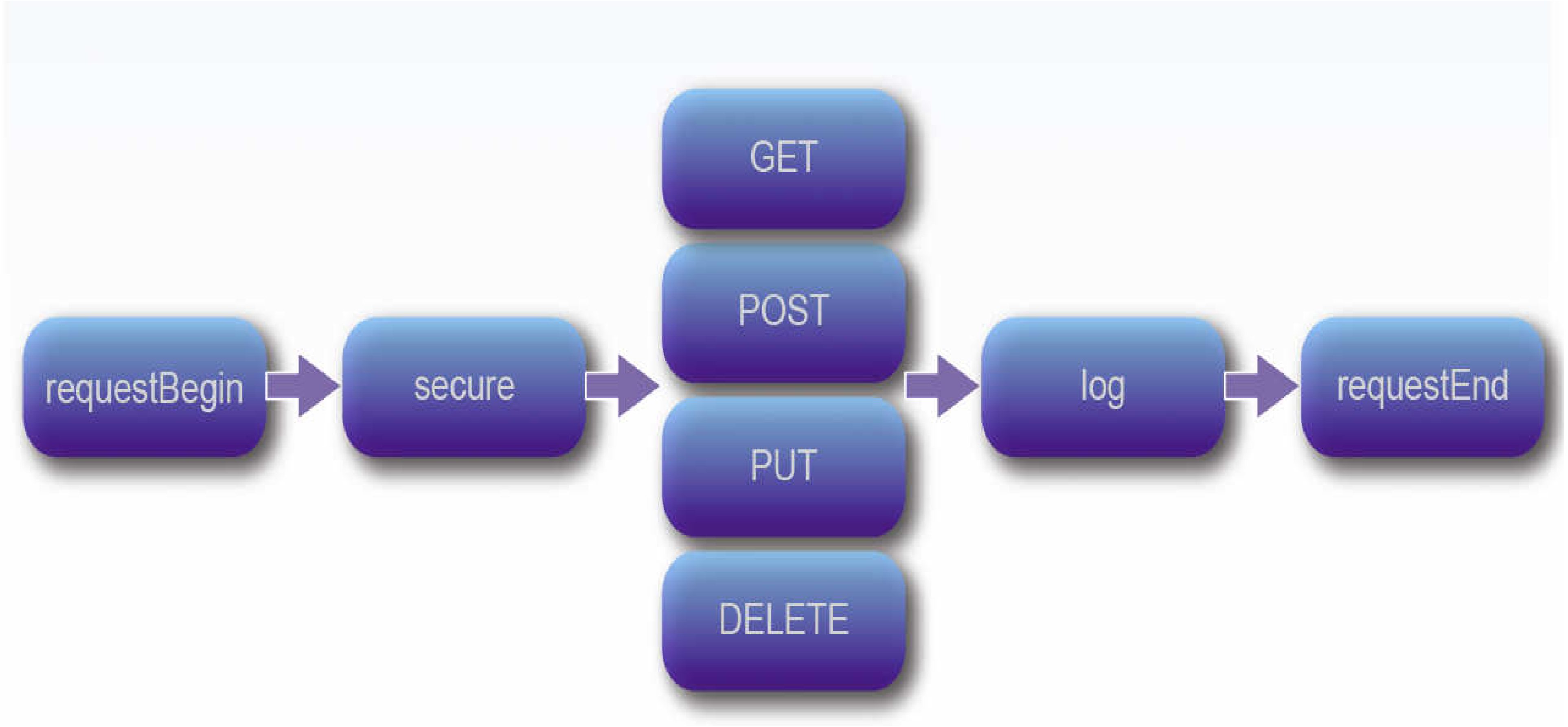
- Simple, free registration process

* Requires acceptance of an IBM license agreement

Core Programming Model

Events

- All behavior in the system is modeled as a set of event
  - Applications are built by handling these events and providing desired behavior
  - Similar to AJAX model or classic UI programming



Event Handlers

All handlers are stateless

Can be implemented in Groovy, PHP, and Java

Groovy

```
println "Hello World"
```

```
def onGET()
{
   println "Hello World"
}
```

PHP

```
function onGET()
{
   echo "Hello World";
}
```

Java

```
public void onGET()
{
   PrintWriter writer = (PrintWriter)zget("/request/writer");
   writer.println("Hello World");
}
```

```
/config/handlers += {
   "events" : "GET",
   "handler" : "HelloWorld.class",
   "conditions" : ["/request/path matches /hello"]
}
```

Global Context – State Management

The Global Context (GC) provides access to and management of all application state

- Conceptually a map of data

Externalizes all state from the application logic

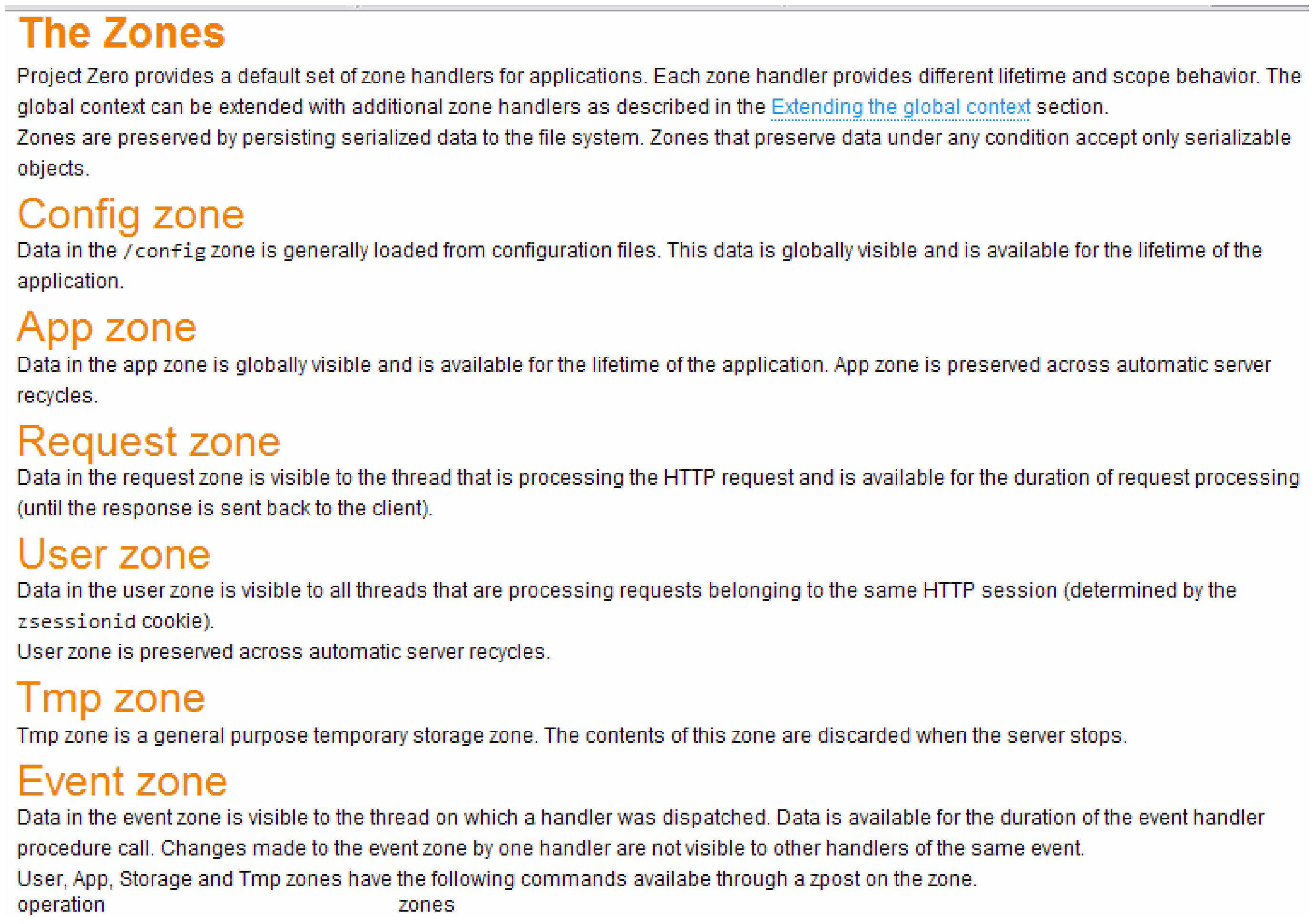
- Enables the restart ability of the JVM without data loss
- Enables clustering and scaling to be added transparently

Simplifies and unifies access to application state and data structures and simplifies state passing within the application

Contains information provided by both the runtime (such as request parameters) and by the application

Global Context Zone

Divided into zones representing different data lifecycles



The Zones

Project Zero provides a default set of zone handlers for applications. Each zone handler provides different lifetime and scope behavior. The global context can be extended with additional zone handlers as described in the Extending the global context section.

Zones are preserved by persisting serialized data to the file system. Zones that preserve data under any condition accept only serializable objects.

Config zone

Data in the `/config` zone is generally loaded from configuration files. This data is globally visible and is available for the lifetime of the application.

App zone

Data in the app zone is globally visible and is available for the lifetime of the application. App zone is preserved across automatic server recycles.

Request zone

Data in the request zone is visible to the thread that is processing the HTTP request and is available for the duration of request processing (until the response is sent back to the client).

User zone

Data in the user zone is visible to all threads that are processing requests belonging to the same HTTP session (determined by the `zsessionid` cookie).

User zone is preserved across automatic server recycles.

Tmp zone

Tmp zone is a general purpose temporary storage zone. The contents of this zone are discarded when the server stops.

Event zone

Data in the event zone is visible to the thread on which a handler was dispatched. Data is available for the duration of the event handler procedure call. Changes made to the event zone by one handler are not visible to other handlers of the same event.

User, App, Storage and Tmp zones have the following commands availabe through a zpost on the zone.

operation	zones

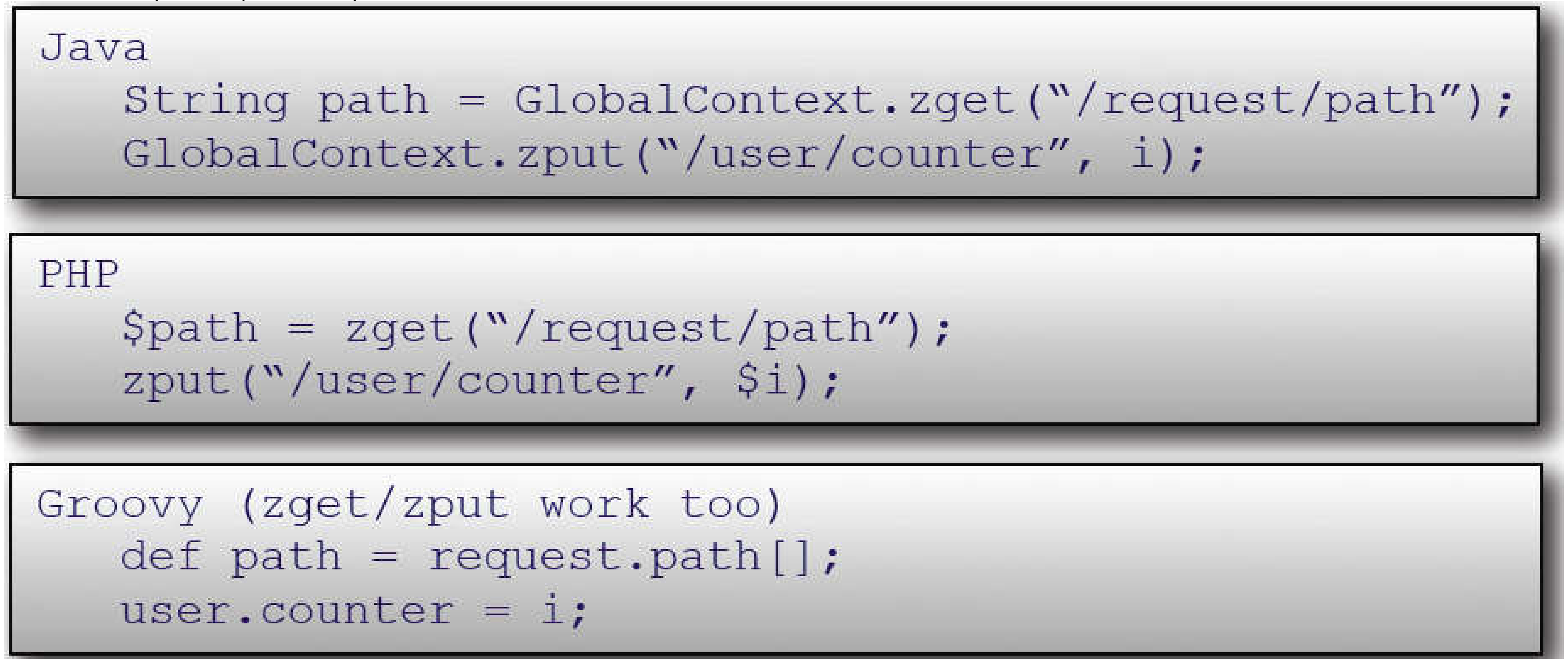
Accessing the Global Context

Data is organized by a URI structure

- First part of URI is always the Zone name
  - /app, /user, /request, /config, /event, /client

Access is modeled after REST

- GET, PUT, POST, DELETE



```
Java
    String path = GlobalContext.zget("/request/path");
    GlobalContext.zput("/user/counter", i);
```

```
PHP
    $path = zget("/request/path");
    zput("/user/counter", $i);
```

```
Groovy (zget/zput work too)
    def path = request.path[];
    user.counter = i;
```

Value Pathing

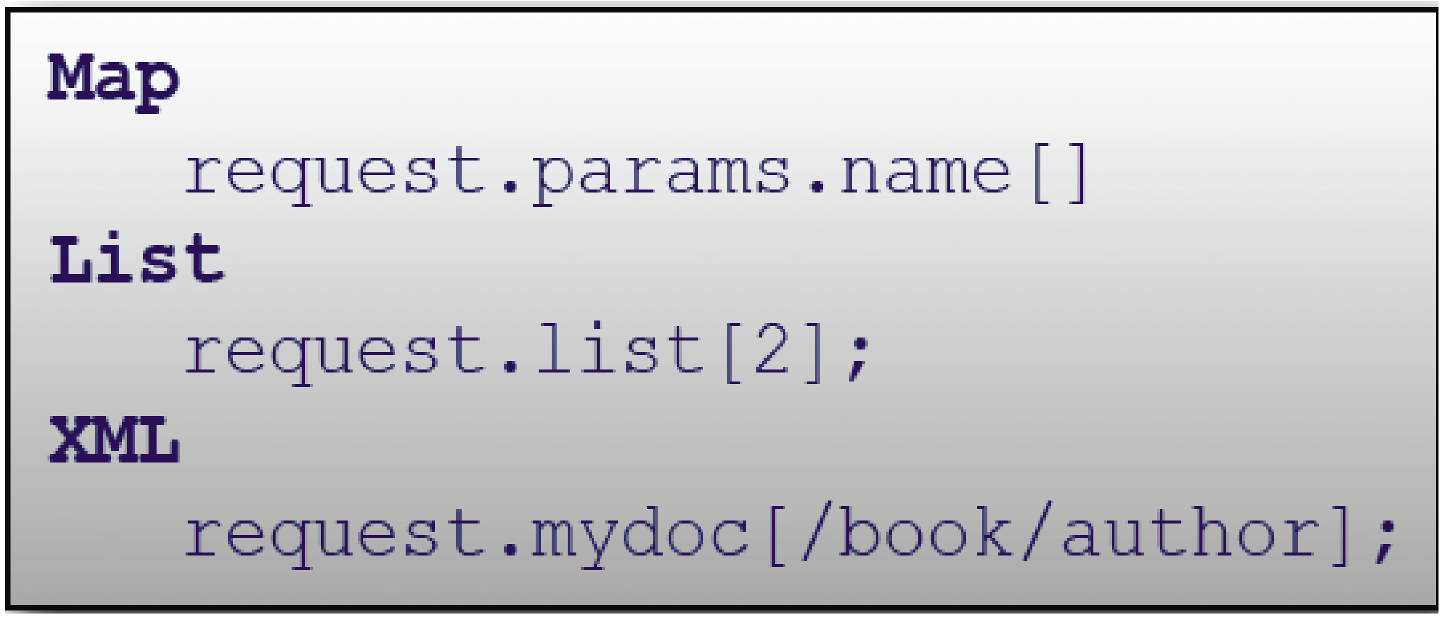
The GC provides simplified access to certain data structures

- Called ***Value Pathing***

Understands

- Maps, List, Objects, XML, JSON

□Allows read and write access to internals of the structure through the GC address



Map

```
    request.params.name[]
```

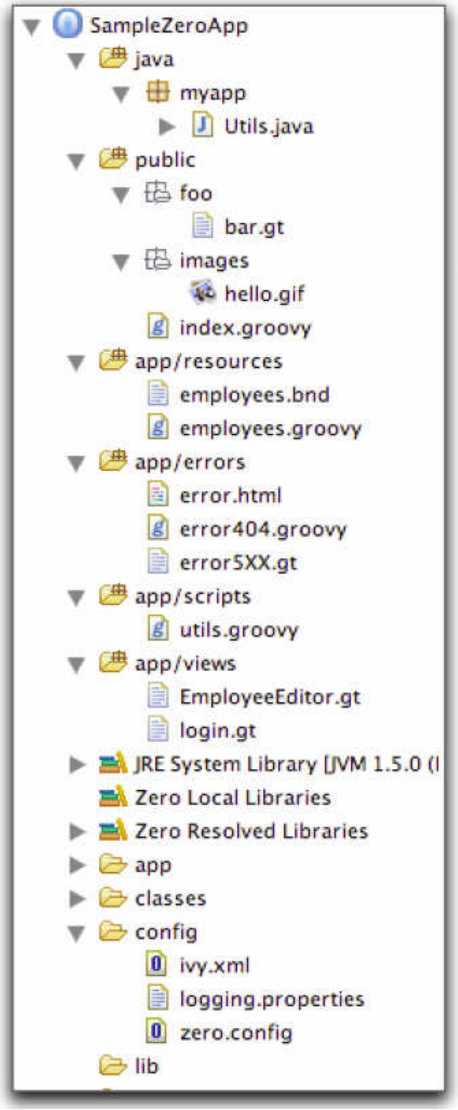
List

```
    request.list[2];
```

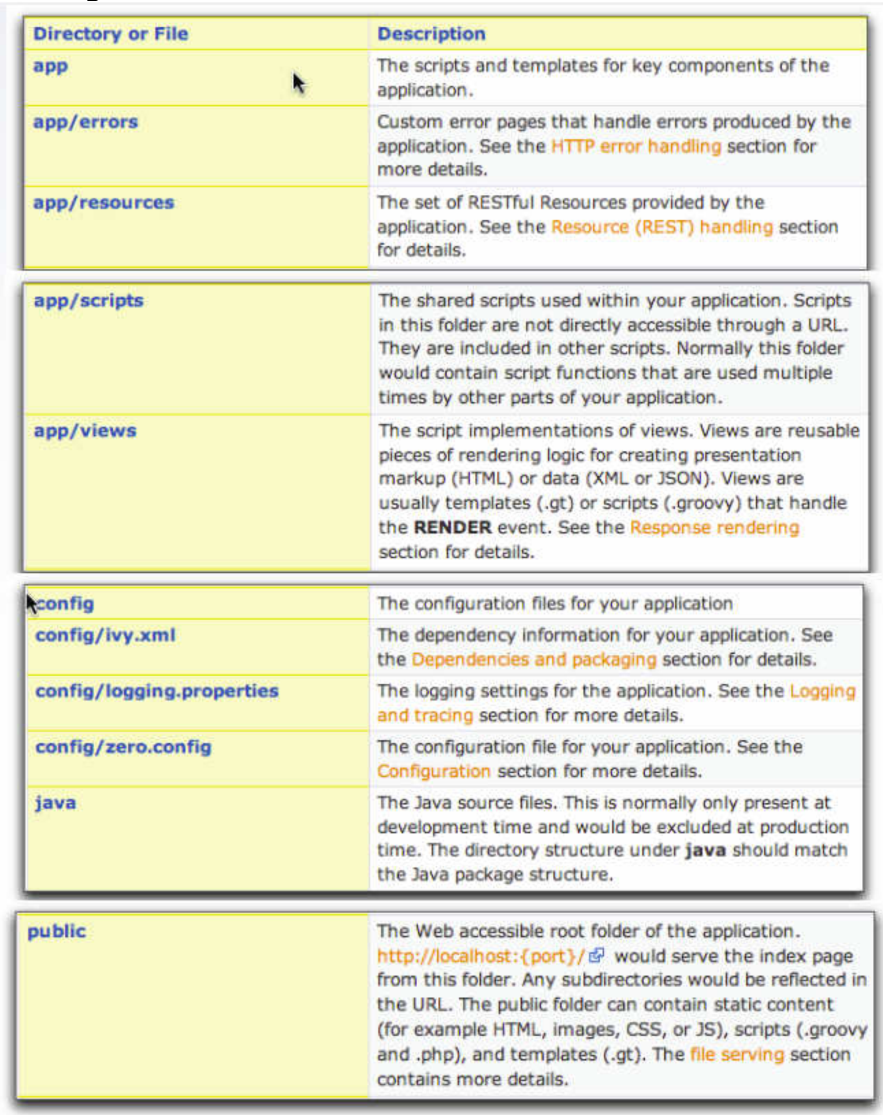
XML

```
    request.mydoc[/book/author];
```

Application Directory Layout



```
▼ SampleZeroApp
    ▼ java
        ▼ myapp
            ▶ Utils.java
    ▼ public
        ▼ foo
            bar.gt
        ▼ images
            hello.gif
        index.groovy
    ▼ app/resources
        employees.bnd
        employees.groovy
    ▼ app/errors
        error.html
        error404.groovy
        error5XX.gt
    ▼ app/scripts
        utils.groovy
    ▼ app/views
        EmployeeEditor.gt
        login.gt
    ▶ JRE System Library [JVM 1.5.0 (I
      Zero Local Libraries
    ▶ Zero Resolved Libraries
    ▶ app
    ▶ classes
    ▼ config
        ivy.xml
        logging.properties
        zero.config
      lib
```

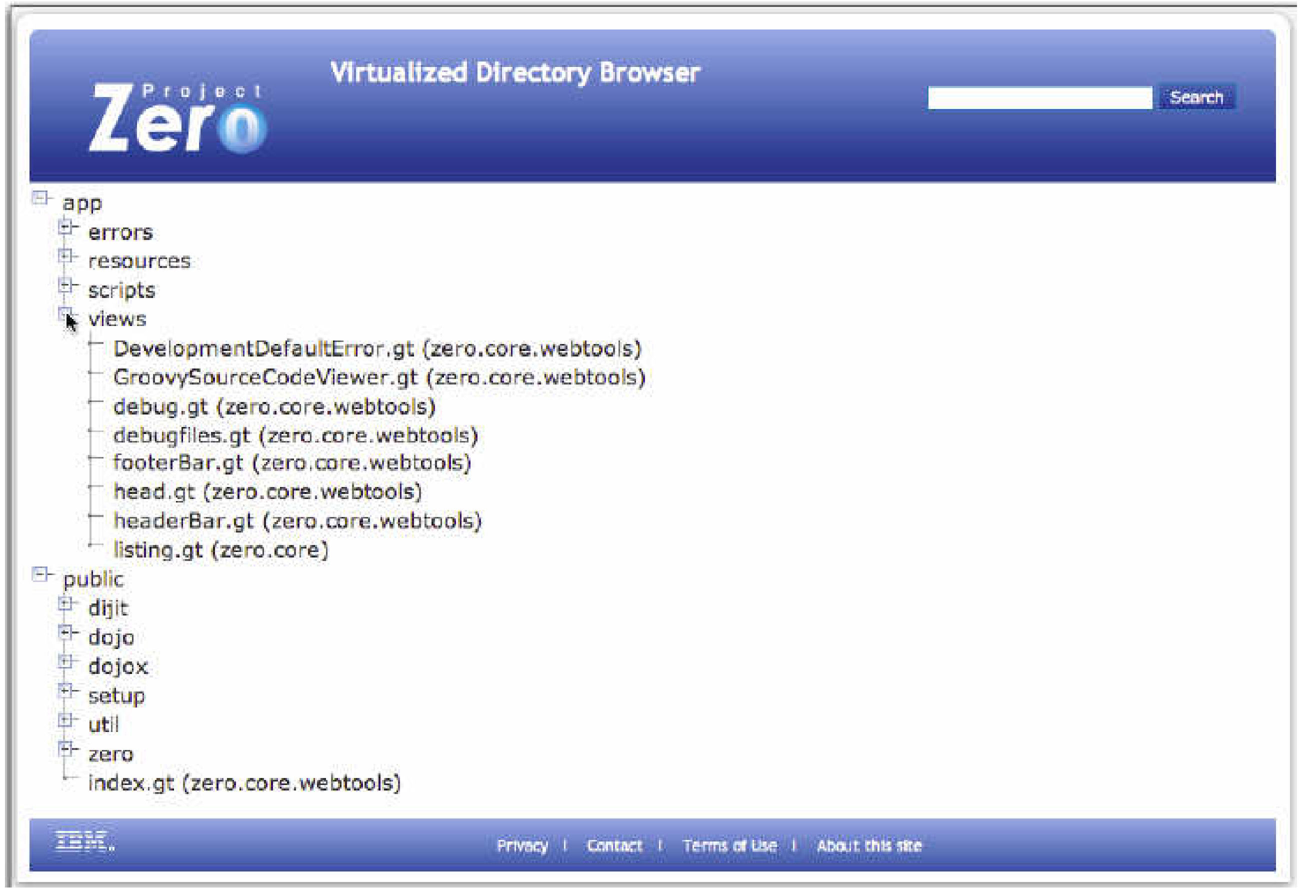


Directory or File	Description
app	The scripts and templates for key components of the application.
app/errors	Custom error pages that handle errors produced by the application. See the HTTP error handling section for more details.
app/resources	The set of RESTful Resources provided by the application. See the Resource (REST) handling section for details.
app/scripts	The shared scripts used within your application. Scripts in this folder are not directly accessible through a URL. They are included in other scripts. Normally this folder would contain script functions that are used multiple times by other parts of your application.
app/views	The script implementations of views. Views are reusable pieces of rendering logic for creating presentation markup (HTML) or data (XML or JSON). Views are usually templates (.gt) or scripts (.groovy) that handle the **RENDER** event. See the Response rendering section for details.
config	The configuration files for your application
config/ivy.xml	The dependency information for your application. See the Dependencies and packaging section for details.
config/logging.properties	The logging settings for the application. See the Logging and tracing section for more details.
config/zero.config	The configuration file for your application. See the Configuration section for more details.
java	The Java source files. This is normally only present at development time and would be excluded at production time. The directory structure under **java** should match the Java package structure.
public	The Web accessible root folder of the application. http://localhost:{port}/ would serve the index page from this folder. Any subdirectories would be reflected in the URL. The public folder can contain static content (for example HTML, images, CSS, or JS), scripts (.groovy and .php), and templates (.gt). The file serving section contains more details.

Virtualized Directory

Project Zero provides seamless integration of directories across an application and its dependencies, while maintaining each as separate entities.

All artifacts are searched within both the application and its declared dependencies


Project Zero
Virtualized Directory Browser
Search
Privacy | Contact | Terms of Use | About this site

```
app
  errors
  resources
  scripts
  views
    DevelopmentDefaultError.gt (zero.core.webtools)
    GroovySourceCodeViewer.gt (zero.core.webtools)
    debug.gt (zero.core.webtools)
    debugfiles.gt (zero.core.webtools)
    footerBar.gt (zero.core.webtools)
    head.gt (zero.core.webtools)
    headerBar.gt (zero.core.webtools)
    listing.gt (zero.core)
public
  dijit
  dojo
  dojox
  setup
  util
  zero
  index.gt (zero.core.webtools)
```

Configuration

Zero configuration file: zero.config

The config/zero.config file is processed at the start of a Zero application.

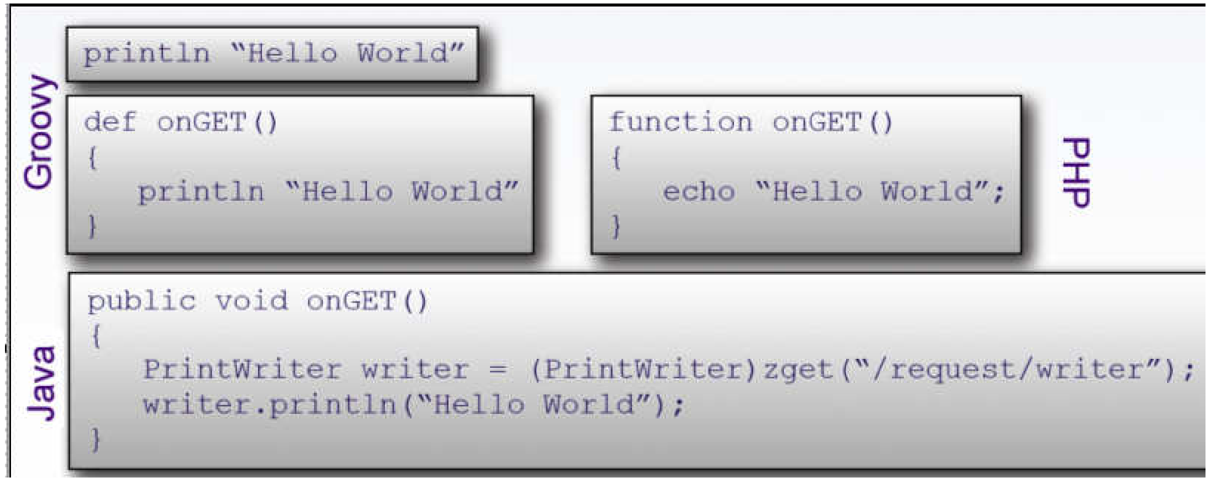
The content of a config/zero.config file is organized into "stanzas" of related key/value pairs. Stanzas are associated with directives, such as "store to the Global Context" and "include another configuration file."

```
# Value set
/config/http/port = 8080

# List set
/config/resources/defaultExtensions = [".groovy"]

# List append
/config/bindings/.groovy +=
["zero.core.groovysupport.bindings.DefaultBindings"]

# Map set
/config/test/map = { "a" : "b", "c" : "d" }

# Map append
/config/test/mapappend += { "a" : "b", "c" : "d" }
/config/test/mapappend += { "x" : "y", "w" : "z" }

# Event handler
/config/handlers += {
  "events" : "GET",
  "handler" : "custom.Handler.class"
}

# Value reference (insert value read at config-load time)
/config/property/myPrefix = "/foo/bar"
/config/test/value = "${/config/property/myPrefix}/bat"

# Variable set/value reference
myPrefix = "/foo/bar"
/config/test/value = "${myPrefix}/bat"

# Include
@include "${/config/dependencies/zero.core}/config/security/
form.config"
{ "formLoginPage" : "/login" }
```

Rendering

Direct

Indirect

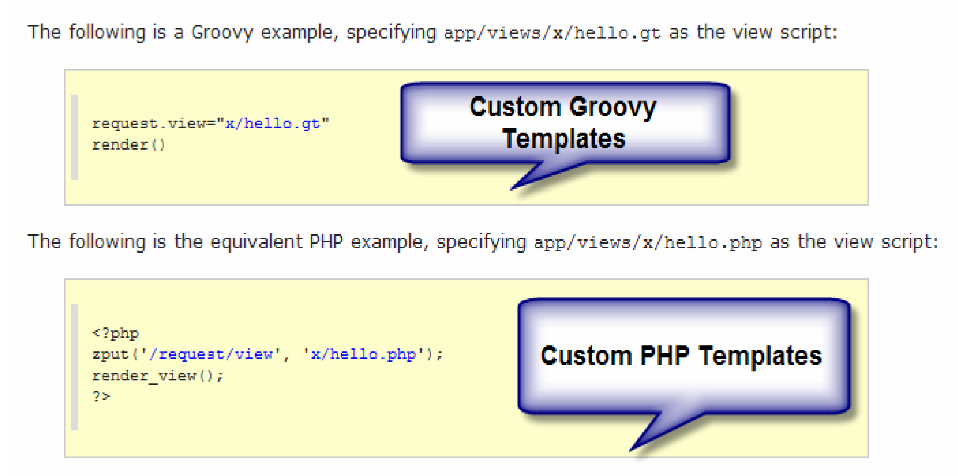


```
println "Hello World"
```

```
def onGET()
{
   println "Hello World"
}
```

```
function onGET()
{
   echo "Hello World";
}
```

```
public void onGET()
{
   PrintWriter writer = (PrintWriter)zget("/request/writer");
   writer.println("Hello World");
}
```

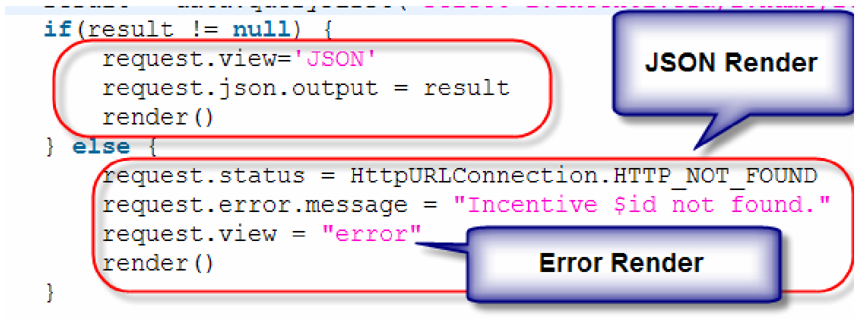


The following is a Groovy example, specifying `app/views/x/hello.gt` as the view script:

```
request.view="x/hello.gt"
render()
```

The following is the equivalent PHP example, specifying `app/views/x/hello.php` as the view script:

```
<?php
zput('/request/view', 'x/hello.php');
render_view();
?>
```



```
if(result != null) {
    request.view='JSON'
    request.json.output = result
    render()
} else {
    request.status = HttpURLConnection.HTTP_NOT_FOUND
    request.error.message = "Incentive $id not found."
    request.view = "error"
    render()
}
```

REST

What is REST?

- Representational State Transfer
- Architectural model on which the World Wide Web is based
- Principles of REST
  - Resource-centric approach
  - All relevant resources are addressable via URIs
  - Uniform access via HTTP – GET, POST, PUT, DELETE
  - Content-type negotiation allows retrieving alternative representations from same URI
- REST-style services
  - are easy to access from code running in web browsers, any other client or servers
  - can serve multiple representations of the same resource

Accessibility for Developers

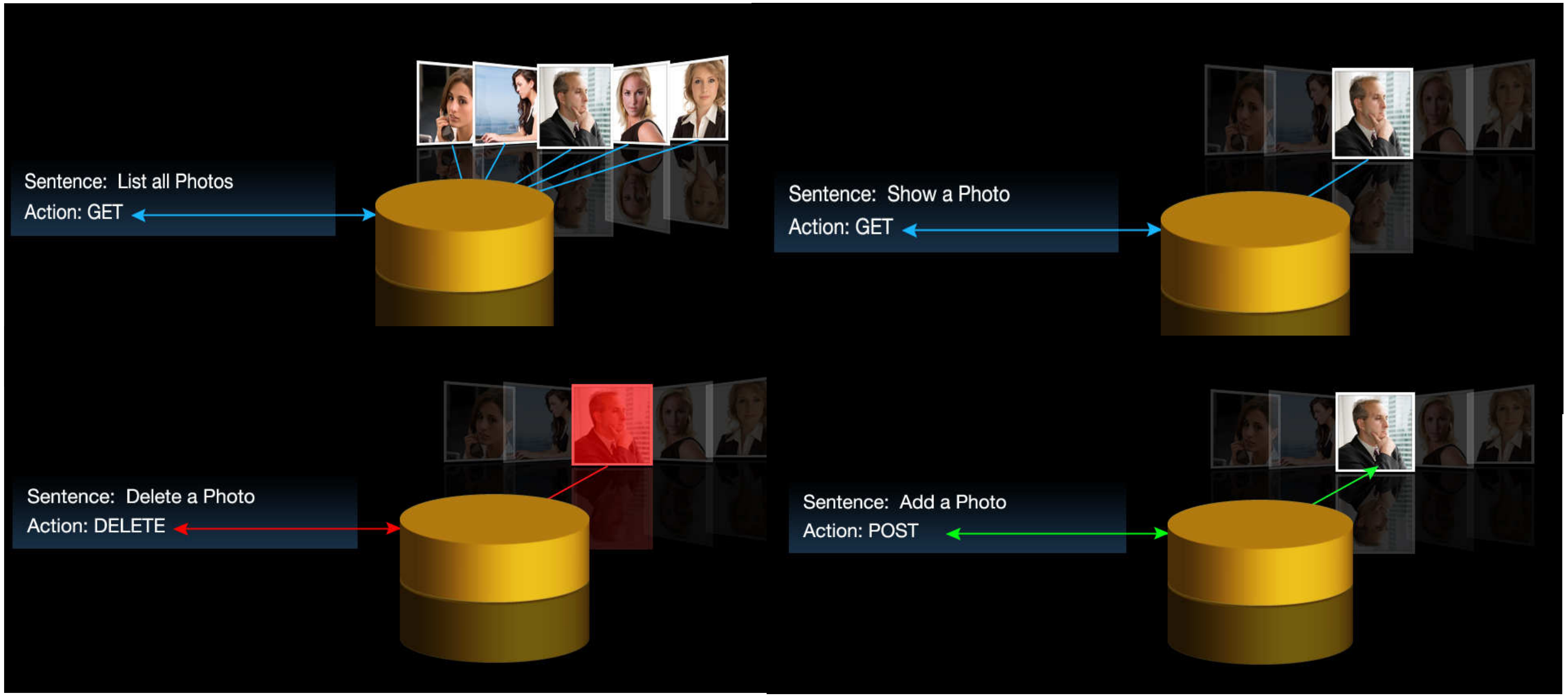
Simply exposing services from the enterprise as URLs and Feeds

A RESTful Web service is formed like a sentence:

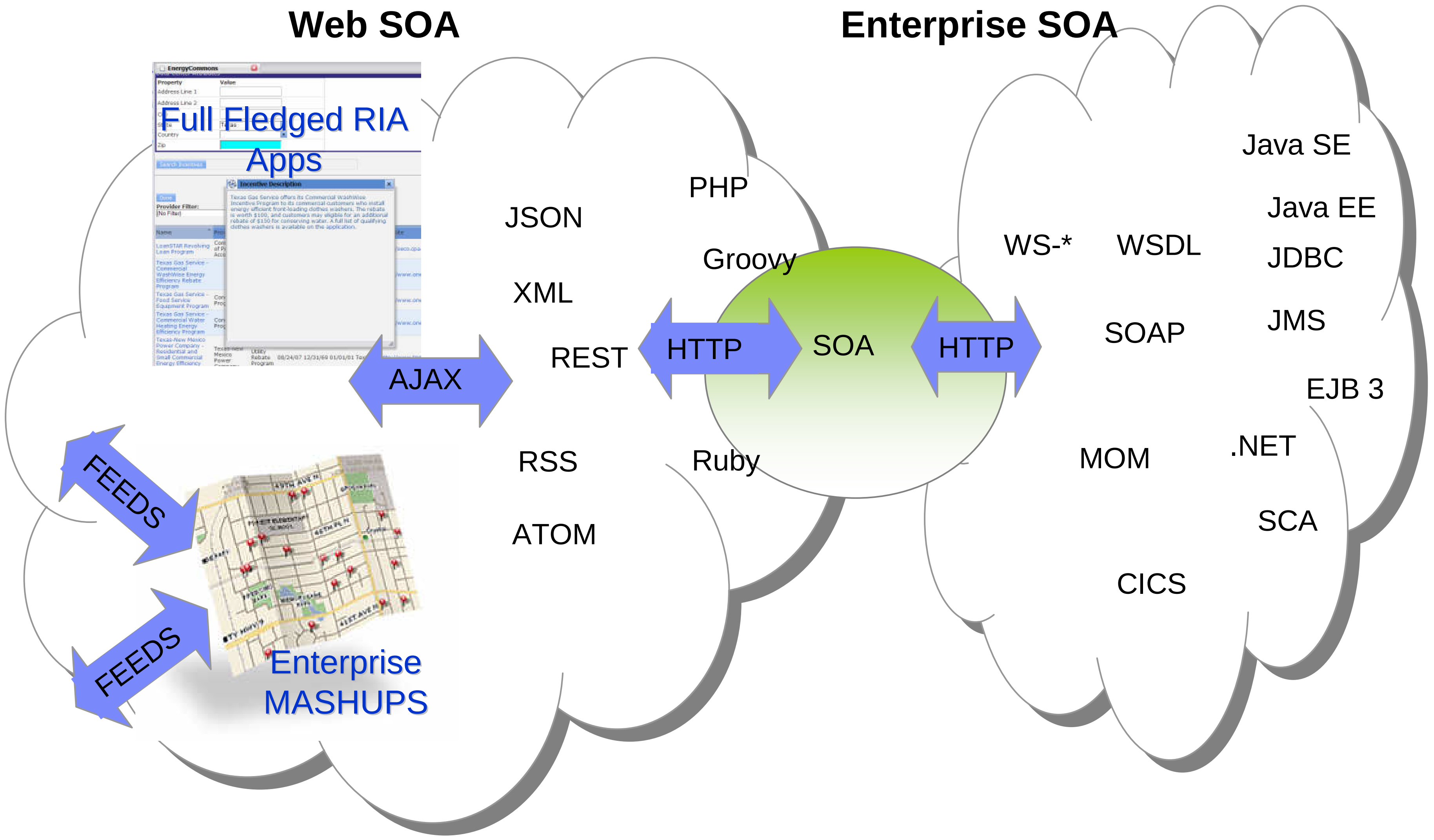
Verb = HTTP Action (GET, POST, PUT, DELETE)

Noun = the URI of the Service (the document)

Adjective = MIME type of the resulting document



Bridging Web SOA and Enterprise SOA



WebSphere sMash and REST

Resources on the Web

What are the URIs?

Which methods are supported at each URI?

What formats?

Resource	**URI**	**Method**	**Representation**	**Description**
Employee list	/resources/employee	GET	JSON (list)	List
		POST	JSON (employee)	Create
Employee	/resources/employee/{id}	GET	JSON (employee)	Retrieve
		PUT	JSON (employee)	Update
		DELETE		Delete

Representations

Employee

```
{
  "first_name“ : "Alice",
  "last_name“  : "Rogers",
  "location"   : "Seattle"
}
```

List of employees

```
[{
  "first_name“ : "Alice",
  "last_name“  : "Rogers",
  "location"   : "Seattle"
 },
 {
  "first_name“ : "Bill”,
  "last_name"  : "Stevens",
  "location"   : "Seattle“
}]
```

Resource Handlers in Zero

Basic event handlers for /resources/*

URI pattern	**Method**	**Event**	**Description**
/resources/collection	GET	list	List of all members
	POST	create	Create member
/resources/collection/{id}	GET	retrieve	Retrieve one member
	PUT	update	Replace member
	DELETE	delete	Delete member

Resource Handlers Example

app/resources/employee.groovy

```
def onList() {
  try {
    // Get configured DataManager for data access
    def data = zero.data.groovy.Manager.create('employee_db')

    // Retrieve employee records via Data Zero
    def result = data.queryArray('SELECT * FROM employees')
    request.view = 'JSON'
    request.json.output = result
    render()

  } catch (Exception e){
    if (e.getCause() instanceof java.sql.SQLException) {
      request.status = HttpURLConnection.HTTP_INTERNAL_ERROR
      request.view = 'error'
      request.error.message = 'The db may not have been initialized.'
      render()
    }
  }
} def onCreate() { ...
```

Resource Handlers Example

app/resources/employee.groovy (continued)

```
def onCreate() {
  // Convert entity to JSON object
  def emp = zero.json.Json.decode(request.input[])

  // Get configured DataManager for data access
  def data = zero.data.groovy.Manager.create('employee_db')

  // Insert employee record via Data Zero APIs
  data.update("""
    INSERT INTO employees
      (username, firstname, lastname, location, phonenumber)
    VALUES ($emp.username, $emp.firstname, $emp.lastname,
            $emp.location, $emp.phonenumber)
""")

  // Set a Location header with URI to the new record
  locationUri = getRequestedUri(false) + '/' + emp['username']
  request.headers.out.Location = locationUri
  request.status = HttpURLConnection.HTTP_NO_CONTENT
}
```

Renderers

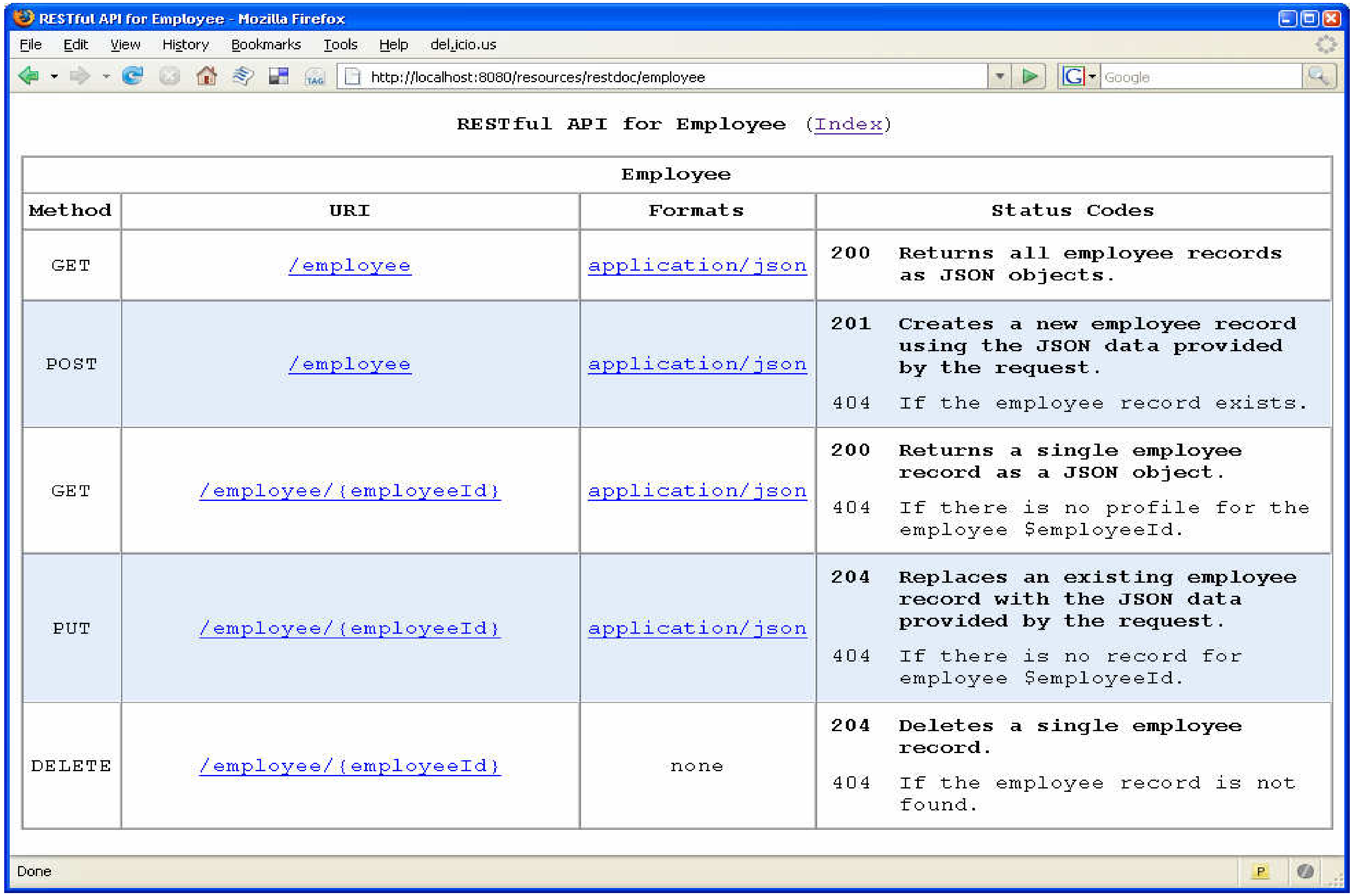
```
request.view = ‘JSON’

request.json.output = object

render()
```

JSON, XML

ATOM

Map	=> Atom entry
List<Map>	=> Atom feed

RESTdoc


RESTful API for Employee - Mozilla Firefox
File Edit View History Bookmarks Tools Help del.icio.us
http://localhost:8080/resources/restdoc/employee
Google
Done

RESTful API for Employee (Index)

Employee				
Method	URI	Formats	Status Codes	
GET	/employee	application/json	200	Returns all employee records as JSON objects.
POST	/employee	application/json	201	Creates a new employee record using the JSON data provided by the request.
			404	If the employee record exists.
GET	/employee/{employeeId}	application/json	200	Returns a single employee record as a JSON object.
			404	If there is no profile for the employee $employeeId.
PUT	/employee/{employeeId}	application/json	204	Replaces an existing employee record with the JSON data provided by the request.
			404	If there is no record for employee $employeeId.
DELETE	/employee/{employeeId}	none	204	Deletes a single employee record.
			404	If the employee record is not found.

An alternative: Zero Resource Model (ZRM)

Model application data

- Constrained set of APIs encourages a **RESTful** application architecture
- Data model that maps well into Atom feeds and JSON formats
- Robust framework for persistence, validation, and serialization

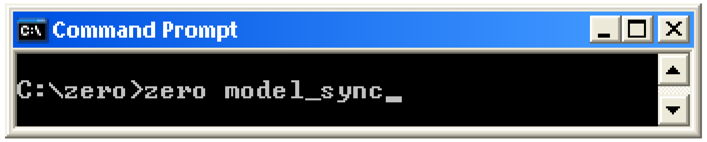
ZRM (continued)

app/models/employee.json

```
{
    "fields" : {
        "first_name": {"type":"string"},
        "last_name": {"type":"string"},
        "location": {"type":"string"}
    }
}
```

app/resources/employee.groovy

```
ZRM.delegate();
```



```
C:\zero>zero model_sync_
```

app/models/fixtures/initial_data.json

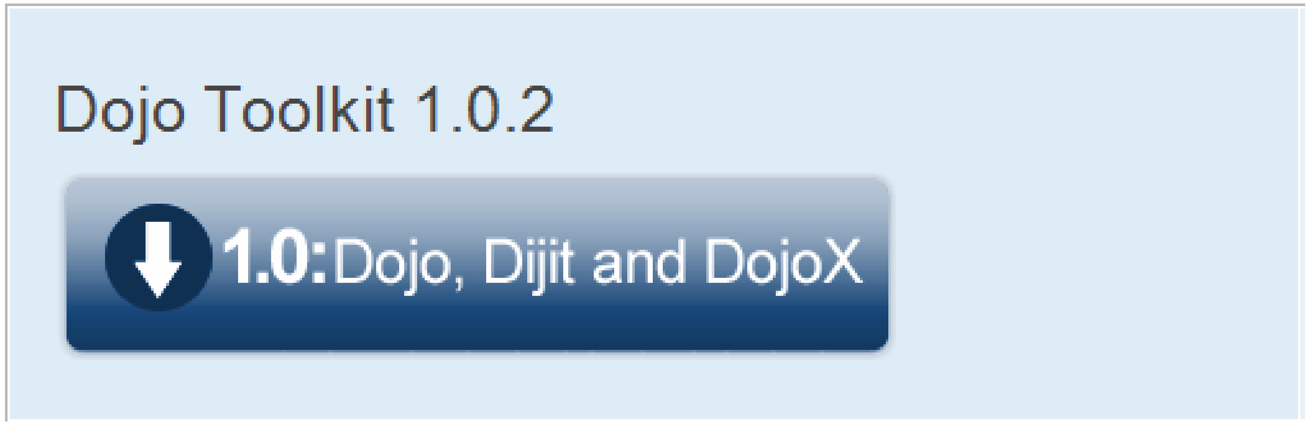
```
[
    {
        "type": "employee",
        "fields": {
                "first_name" :   "Alice",
                "last_name“  :   "Rogers",
                "location"   :   "Seattle"
        }
    },
    {
        "type": "employee",
        "fields": {
                "first_name" :   "Bill",
                "last_name"  :   "Stevens",
                "location"   :   "Seattle"
        }
    },
    {
        "type": "person",
        "fields": {
                "first_name" :   "Cathy",
                "last_name"  :   "Tomlin",
                "location"   :   "Boston"
        }
    }
]
```

WebSphere sMash and RIA

Dojo Broswer Toolkit

Dojo is an Open Source DHTML toolkit written in JavaScript. It builds on several contributed code bases.

- Provides Rich Set of Widgets
- Web UI Framework
- Rich Event handling System
- General Purpose HTML Libraries
- Several other utilities
  - Math, XML to JS parsing, etc…



Dojo Architecture

Base

The kernel of the toolkit wrapped into a **25k** js file (dojo.js). Base bootstraps the toolkit, includes AJAX utilities, class based inheritance, packaging system and more

Core

Provides addition facilities on top of the base for accessing data stores, effects such as wipes/slides, internationalization (i18n) and back-button handling among other things. Separate package keeps base small

Dijit

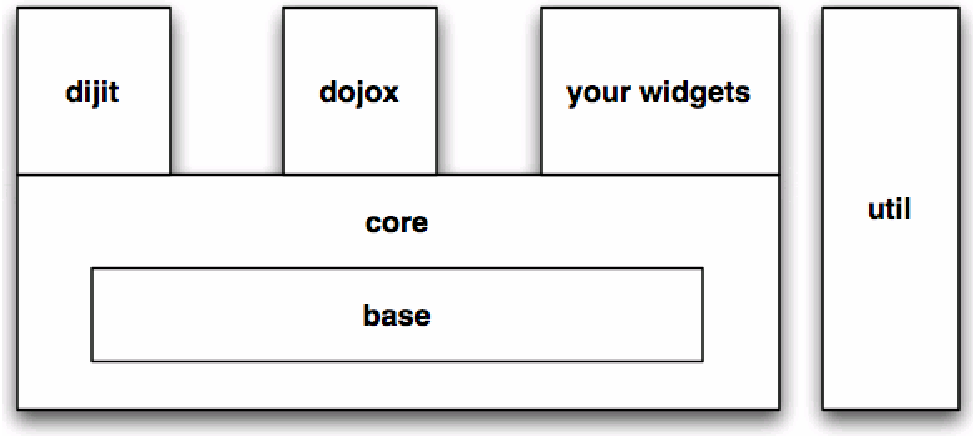
Shorthand for "Dojo widget". Could refer to a single Dojo widget (a dijit) or to the entire component of the toolkit containing all of Dojo's widgets (Dijit)

DojoX

“Dojo Extra" and contains features that stand a chance of one day migrating into Core, Dijit or even a new module. A great proving ground for new features while maintaining standards of core and base.

Util

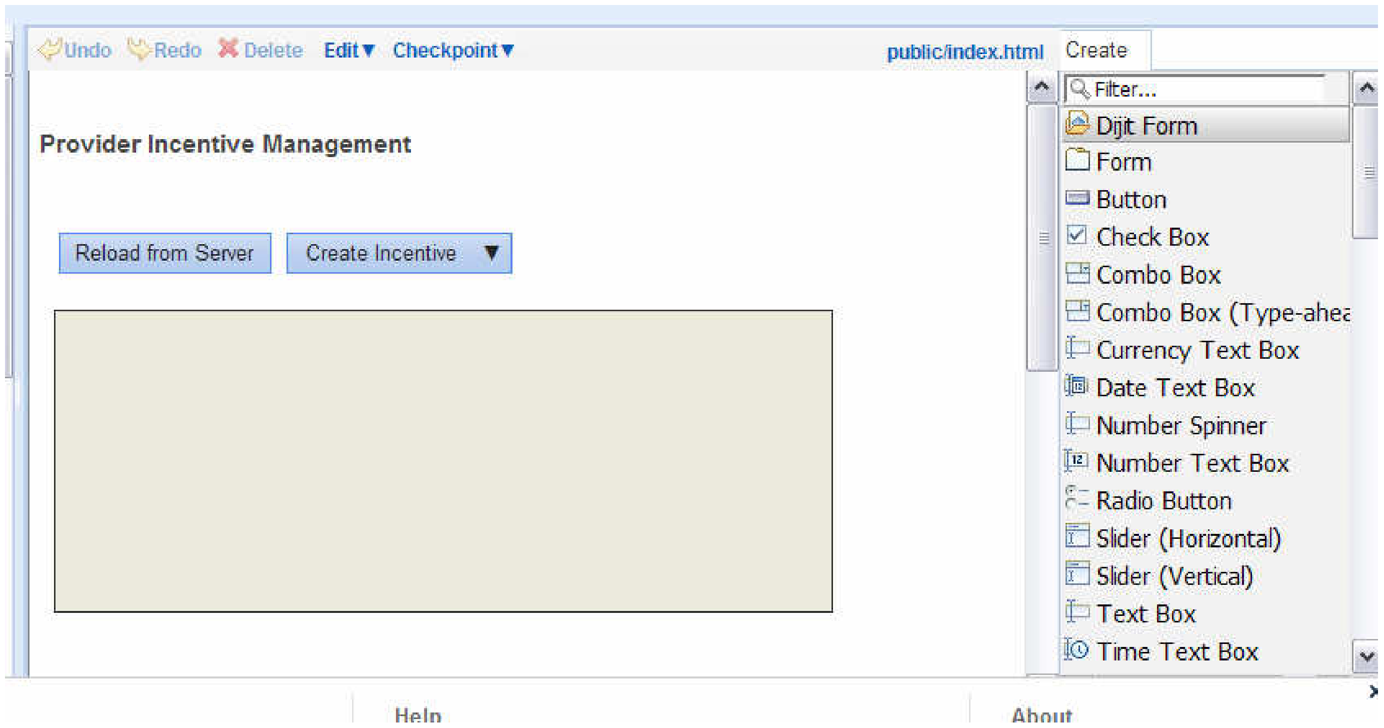
A collection of Dojo utilities (more later)



Web Based IDE Editor for Dojo

Dojo connections with Services through Wires.

Drag and Drop with Dojo Dijits.



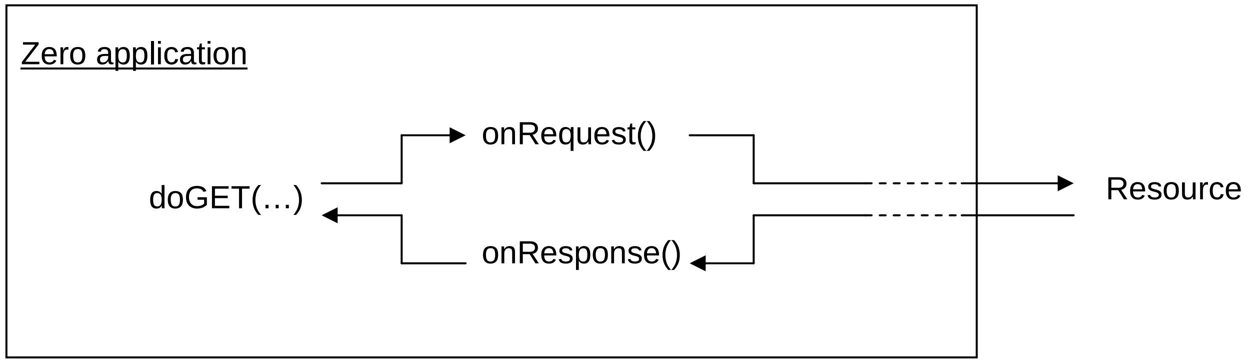
Connections

RESTful API to resources via a variety of protocols

- HTTP/S
- SMTP
- File
- JMS

In-process mediations

- SOAP
- Custom



WebSphere sMash and Flows

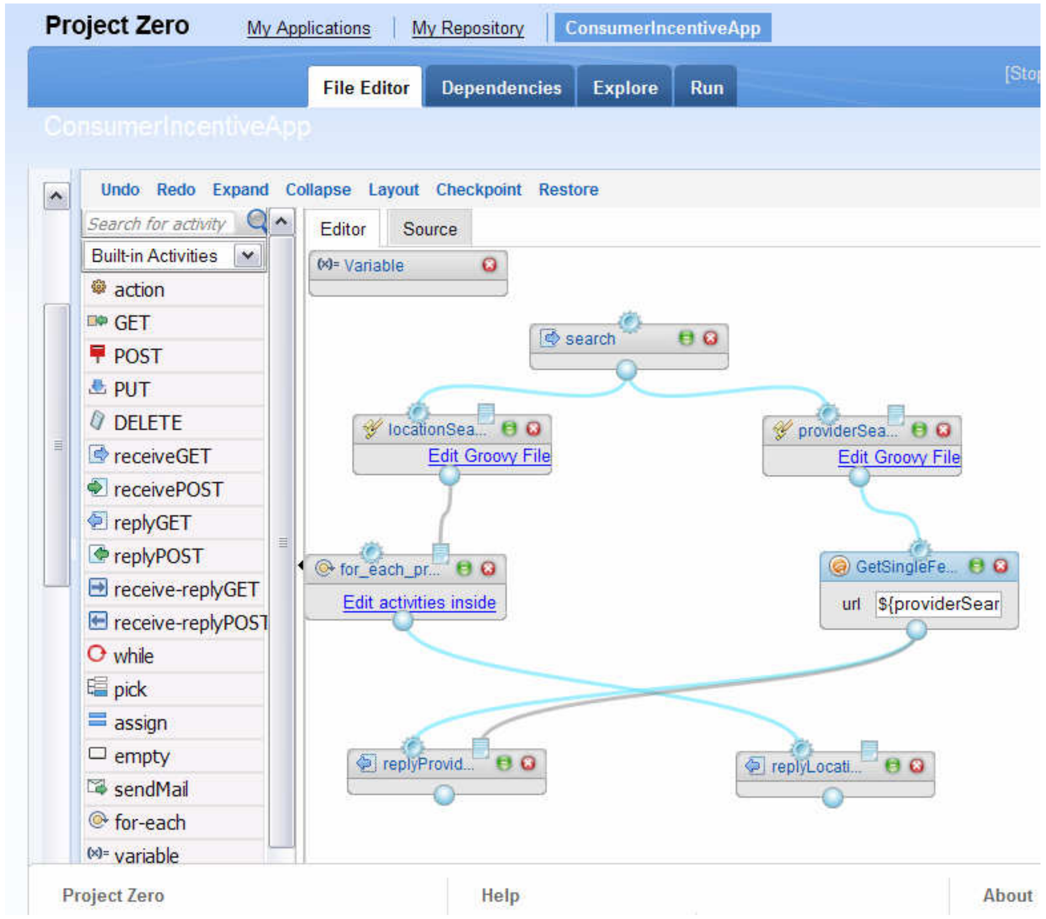
Assemble

Composition of applications by "wiring" REST services using the SPLICE flows. Incorporating both activity and data flows.

A solution may be rapidly assembled by combining existing feeds and services that enrich, sort, and filter data in a pipeline. Either visually or programmatically.

Configure templates to alter pipeline routes, log events along the pipeline, as well as transform data

Adapters to enhance integration with existing systems.



Questions & Answers

AM06

Rational Application Developer v7.5

Geni Hutton

Manager, Rational Application Developer

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

IBM

Rational. software

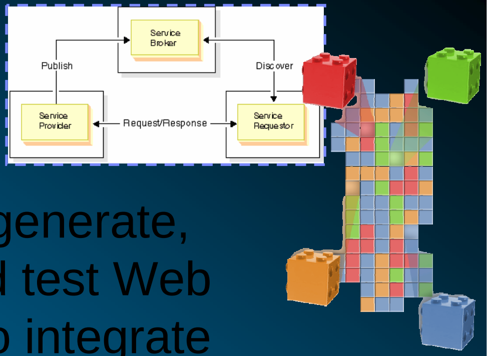
Agenda

- Rational Software Delivery Platform
    - The value RAD adds to your development lifecycle
- Rational Application Developer 7.5
    - EJB 3.0 and JPA
    - Web Tools & JSF Overview
    - Web 2.0 Support
    - Web Services
    - WebSphere support, Server tools, Problem determination
    - WebSphere Portlet and Portal support
    - EIS Adapters
    - Collaboration with other Rational products (RTC, CC plugin…)
- SCA Feature Pack

RAD accelerates development for IBM middleware

Web 2.0

Extend SOA and Java EE assets with dynamic, rich AJAX applications



SOA

Discover, generate, deploy and test Web Services to integrate business applications

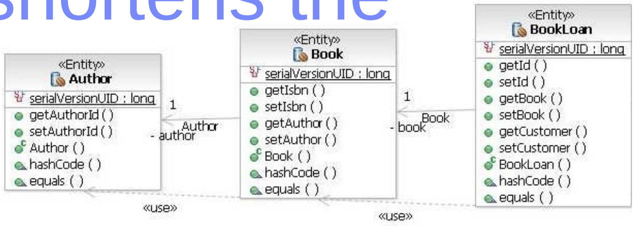
Java EE 5

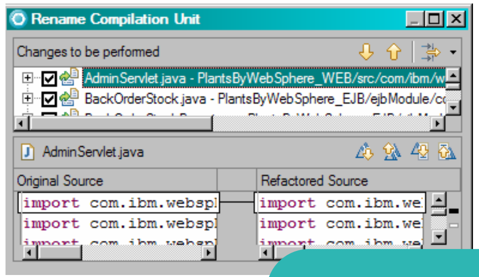
Quickly develop and test Java EE 5 applications, with annotation based programming and integrated WebSphere support

Portal

Rapid visual design of portal and portlets, and testing with WebSphere Portal

RAD increases efficiency and shortens the development and test cycle



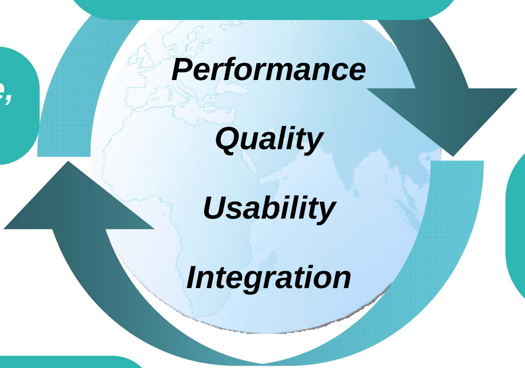

Rename Compilation Unit
Changes to be performed
AdminServlet.java - PlantsByWebSphere_WEB/src/com/ibm/w
BackOrderStock.java - PlantsByWebSphere_EJB/ejbModule/co
AdminServlet.java
Original Source
Refactored Source

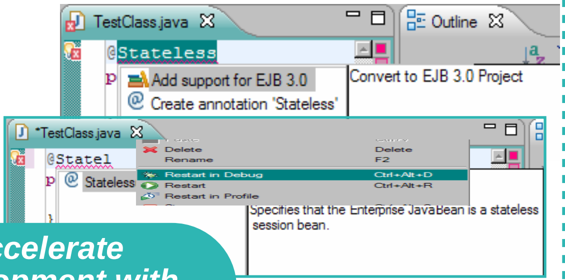
```
import com.ibm.websp
import com.ibm.websp
```

```
import com.ibm.we
import com.ibm.we
```

Visualize and enforce Java EE architectures

Refactor, rename, enhance



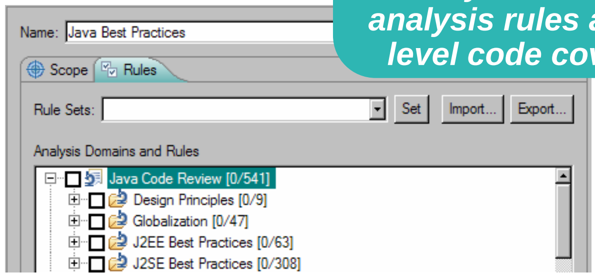


```
@Stateless
```

```
@Statel
```

Specifies that the Enterprise JavaBean is a stateless session bean.

Accelerate development with annotation based programming and quickfixes



Name: Java Best Practices

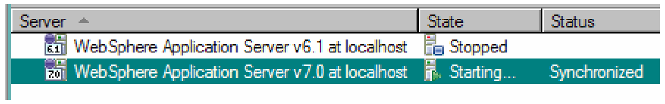
Rule Sets:

Analysis Domains and Rules

- Java Code Review [0/541]
  - Design Principles [0/9]
  - Globalization [0/47]
  - J2EE Best Practices [0/63]
  - J2SE Best Practices [0/308]

Analyze with static analysis rules and line level code coverage

Unit test with WebSphere Integrations



Server	State	Status
WebSphere Application Server v6.1 at localhost	Stopped	
WebSphere Application Server v7.0 at localhost	Starting...	Synchronized

Agenda

- Rational Software Delivery Platform
  - The value RAD adds to your development lifecycle

- Rational Application Developer 7.5
  - EJB 3.0 and JPA
  - Web Tools & JSF Overview
  - Web 2.0 Support
  - Web Services
  - WebSphere support, Server tools, Problem determination
  - WebSphere Portlet and Portal support
  - EIS Adapters
  - Collaboration with other Rational products (RTC, CC plugin…)

Rational Application Developer v7.5 Value Statements

- **Increased developer productivity**
  - Improved iterative development – focus on creation, validation, refactoring and deletion of artifacts (exploit the annotation based programming model style)
  - Programming model support for WAS V7.0 standards support:
    - JEE5 (EJB3.0, JPA, JSF 1.2, JAX-WS 2.0, JAX-B2.0, JSP 2.1, Servlet 2.5, ..
  - Simplified development of RIA clients to extend and expose services and feeds
- **Improved application quality**
  - Line level code coverage, advanced code review and debug capabilities
- **Integration with other IBM products**
  - WebSphere test environments for WAS 7.0, WAS 6.1, and WAS 6.0 included
  - WebSphere feature pack support (Web 2.0 FEP, Web Services FEP, EJB 3.0 FEP)
  - WebSphere Portal 6.1 / 6.0 development (WP 6.1 server included)
  - WebSphere adapter Support (SAP, PeopleSoft, Siebel, ..)
- **Provides governance support**
  - Install Manager allows flexible installation and maintenance
  - Process Advisor guides developers for best of breed practice
- **IBM support**
  - 24x7 phone support
  - Bug fixes

Flexible Install Options

- Rational Application Developer uses IBM Install manager technology to simplify and speed up the install process by only installing the install options chosen by the user
- RAD is built on top of Eclipse 3.4 and uses the IBM JRE 1.6
- Download & footprint improvements for the WebSphere test environment;
  - Support for “base” servers (WebSphere App server + fix pack level)
  - Support for “enhanced” servers (WebSphere App server + fix pack level + feature packs)
  - Build your own server to match your product environment
- Ability to shell share with other Rational products which helps developers manage the lifecycle of their applications on their desktop
- Enterprise install capabilities available to ease the install onto multiple desktops
- Ability to install documentation on a common web server or use the documentation that is available on the web to have access to the latest updates

Agenda

- Rational Software Delivery Platform
  - The value RAD adds to your development lifecycle
- RAD v7.5 enhancements
  - JEE5:EJB 3.0 and JPA
  - Web Tools & JSF Overview
  - JSF Overview
  - Web 2.0 Support
  - Web Services
  - WebSphere support, Server tools, Problem determination
  - WebSphere Portlet and Portal support
  - EIS Adapters
  - Collaboration with other Rational products (RTC, CC plugin…)

What's new with Java EE 5?

- The Java EE 5 platform introduces a simplified programming model.
- Information is inserted as an *annotation* directly into your Java source file.
- Annotations are generally used to embed in a program data that would otherwise be furnished in a side file.
- With annotations, you put the specification information right in your code next to the program element that it affects. This is a more intuitive and convenient approach.   For example:

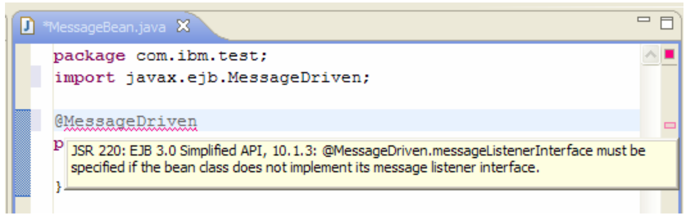
```
@Stateless
public class AccountBean implements IAccount {
}
```

- Developers can use annotations instead of XML deployment descriptors
  - Annotations are better during application development
  - The separate "Deployment Descriptor" side files are now optional, and are better for production deployment (allowing changes without source updates)

RAD helps with: Java EE 5 development

RAD helps simplify and accelerate Java EE 5 development

- Content assist and as you type validation

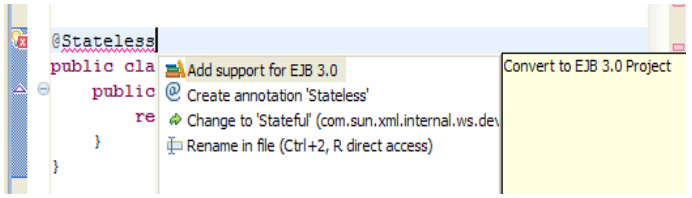


*MessageBean.java

```
package com.ibm.test;
import javax.ejb.MessageDriven;

@MessageDriven
p
}
```

JSR 220: EJB 3.0 Simplified API, 10.1.3: @MessageDriven.messageListenerInterface must be specified if the bean class does not implement its message listener interface.

- Quickfixes for code and project configuration

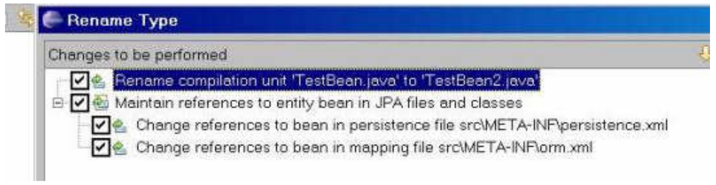


```
@Stateless
public cla
    public
        re
    }
}
```

Add support for EJB 3.0
Create annotation 'Stateless'
Change to 'Stateful' (com.sun.xml.internal.ws.dev
Rename in file (Ctrl+2, R direct access)

Convert to EJB 3.0 Project

- Advanced refactoring options to allow you to modify and maintain code in an iterative manner.

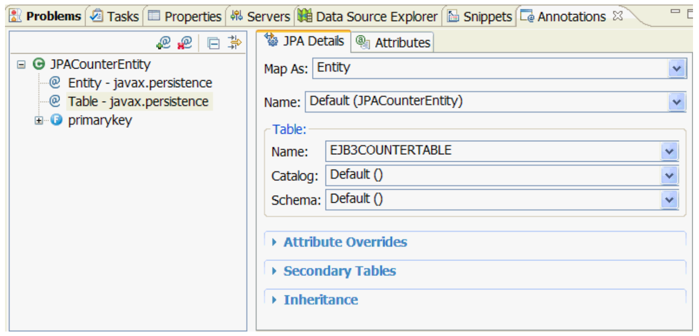


Rename Type

Changes to be performed

- [✓] Rename compilation unit 'TestBean.java' to 'TestBean2.java'
- [✓] Maintain references to entity bean in JPA files and classes
  - [✓] Change references to bean in persistence file src\META-INF\persistence.xml
  - [✓] Change references to bean in mapping file src\META-INF\orm.xml

- Annotation view to manage and modify annotation properties



JPACounterEntity
- Entity - javax.persistence
- Table - javax.persistence
- primarykey

Map As: Entity

Name: Default (JPACounterEntity)

Table:

Name: EJB3COUNTERTABLE

Catalog: Default ()

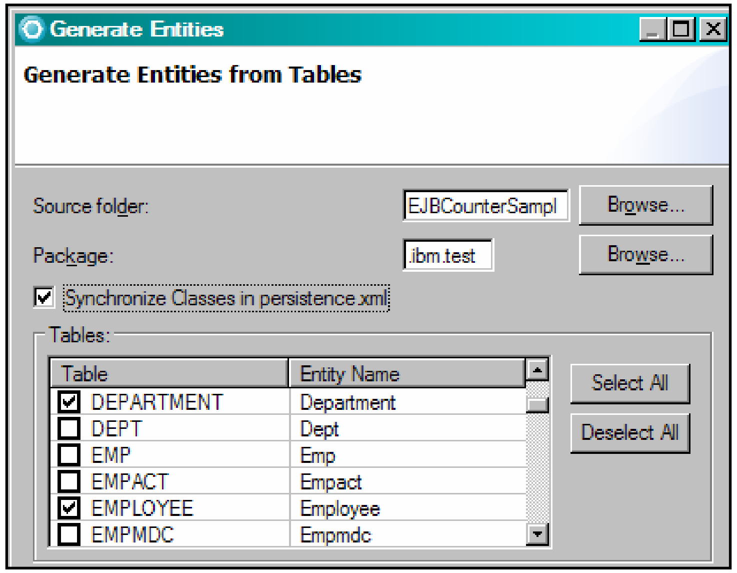
Schema: Default ()

Attribute Overrides

Secondary Tables

Inheritance

RAD helps with: tools to map data to a JPA bean



Generate Entities

Generate Entities from Tables

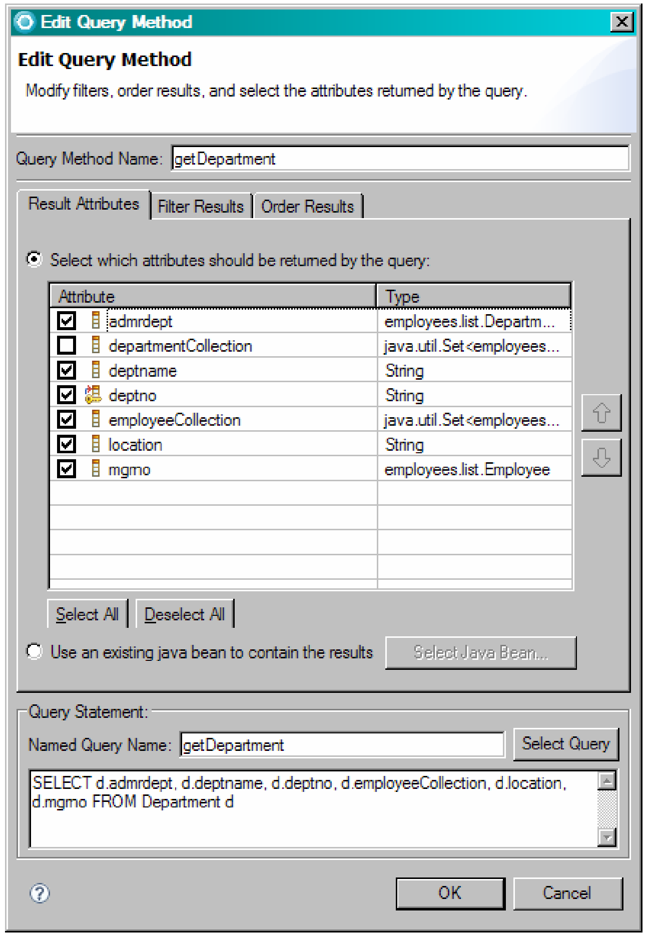
Source folder:	EJBCounterSampl
Package:	ibm.test

[✓] Synchronize Classes in persistence.xml

Tables:

Table	Entity Name
✓ DEPARTMENT	Department
☐ DEPT	Dept
☐ EMP	Emp
☐ EMPACT	Empact
✓ EMPLOYEE	Employee
☐ EMPMDC	Empmdc

Generate entities



Edit Query Method

Edit Query Method

Modify filters, order results, and select the attributes returned by the query.

Query Method Name: getDepartment

(●) Select which attributes should be returned by the query:

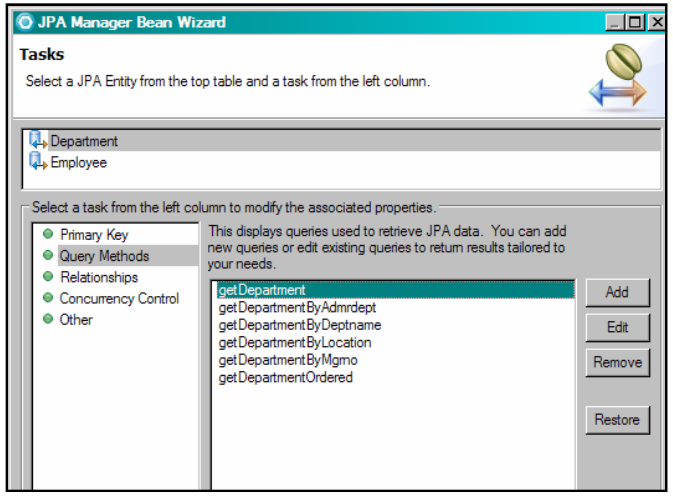
Attribute	Type
✓ admrdept	employees.list.Departm...
☐ departmentCollection	java.util.Set<employees...
✓ deptname	String
✓ deptno	String
✓ employeeCollection	java.util.Set<employees...
✓ location	String
✓ mgrno	employees.list.Employee

( ) Use an existing java bean to contain the results

Query Statement:

Named Query Name: getDepartment

```
SELECT d.admrdept, d.deptname, d.deptno, d.employeeCollection, d.location,
d.mgrno FROM Department d
```



JPA Manager Bean Wizard

Tasks

Select a JPA Entity from the top table and a task from the left column.

- Department
- Employee

Select a task from the left column to modify the associated properties.

This displays queries used to retrieve JPA data. You can add new queries or edit existing queries to return results tailored to your needs.

- getDepartment
- getDepartmentByAdmrdept
- getDepartmentByDeptname
- getDepartmentByLocation
- getDepartmentByMgrno
- getDepartmentOrdered

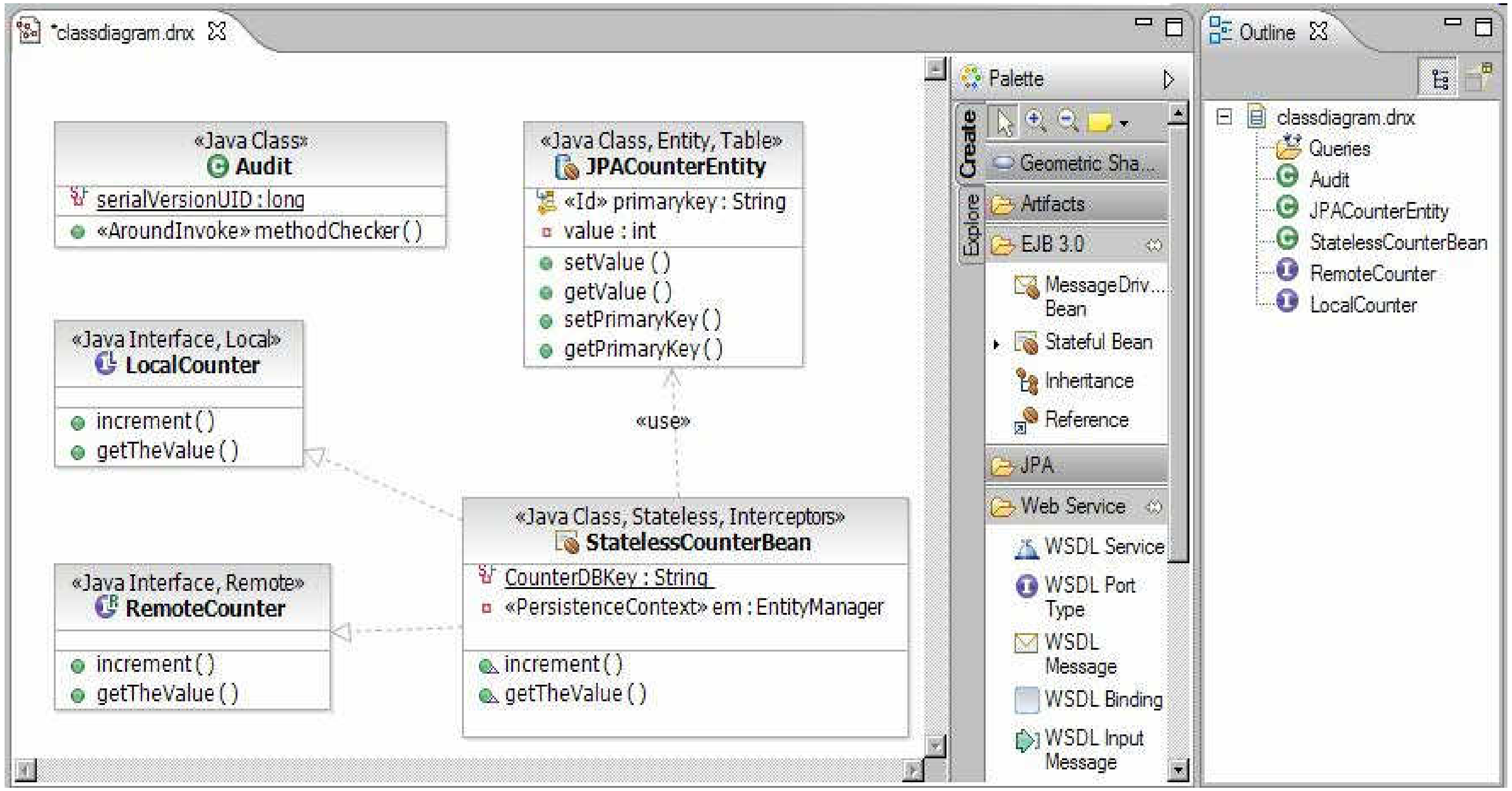
Create Manager Beans

Filter results

RAD helps with: Visualizing your EJB 3.0 beans

EJB Visualizer updated to view & edit EJB 3.0 beans

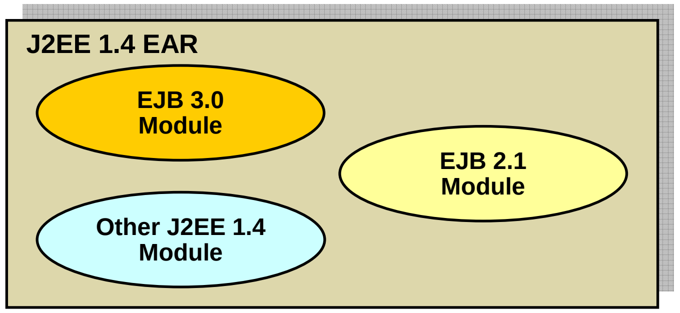
- Beans can be annotation based, or use XML deployment descriptors



RAD helps with: Supported runtime configurations for EJB3

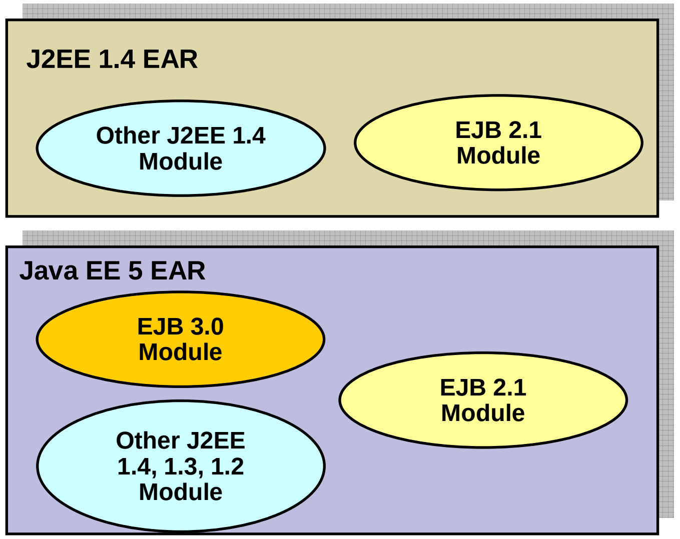
WebSphere Application Server 6.1

- Feature Pack for EJB 3.0
  - Supports EJB 3.0 and JPA specifications
  - EARs can contain EJB 3.0 and J2EE 1.4 modules
  - EAR without a DD can be deployed
  - No EE 5 Web or application client modules



WebSphere Application Server 7.0

- J2EE 1.4 EARs can not contain Java EE 5 modules
- Java EE 5 EARs can contain legacy J2EE (1.4 , 1.3, 1.2) modules.



Agenda

- Rational Software Delivery Platform
  - The value RAD adds to your development lifecycle
- RAD v7.5 enhancements
  - JEE5:EJB 3.0 and JPA
  - Web Tools & JSF Overview
  - Web 2.0 Support
  - Web Services
  - WebSphere support, Server tools, Problem determination
  - WebSphere Portlet and Portal support
  - EIS Adapters
  - Collaboration with other Rational products (RTC, CC plugin…)

What's new - Web Tools

- JPA consumption in Web Applications
- Page Designer
  - Source page includes Significantly enhanced JavaScript support
  - Split view option
    - Ability to split the page designer into a designer and source view
    - Can split view Horizontally or vertically
  - Absolute positioning
    - Positioning elements to an absolute position, via the design page.
    - The tool will apply the CSS style “position:absolute” to layout elements on the page.
- Support for Struts 1.2, 1.3
- JSF 1.2 support
- Integration of third party JSF libraries.
  - Tools to import/manage libraries and add to the Page Designer palette
- Custom Component Library Builder.
  - Allow users to build a JSF component library from existing components and integrate into the tools.
- JWL widget library: enhancements to make library compatible with dojo

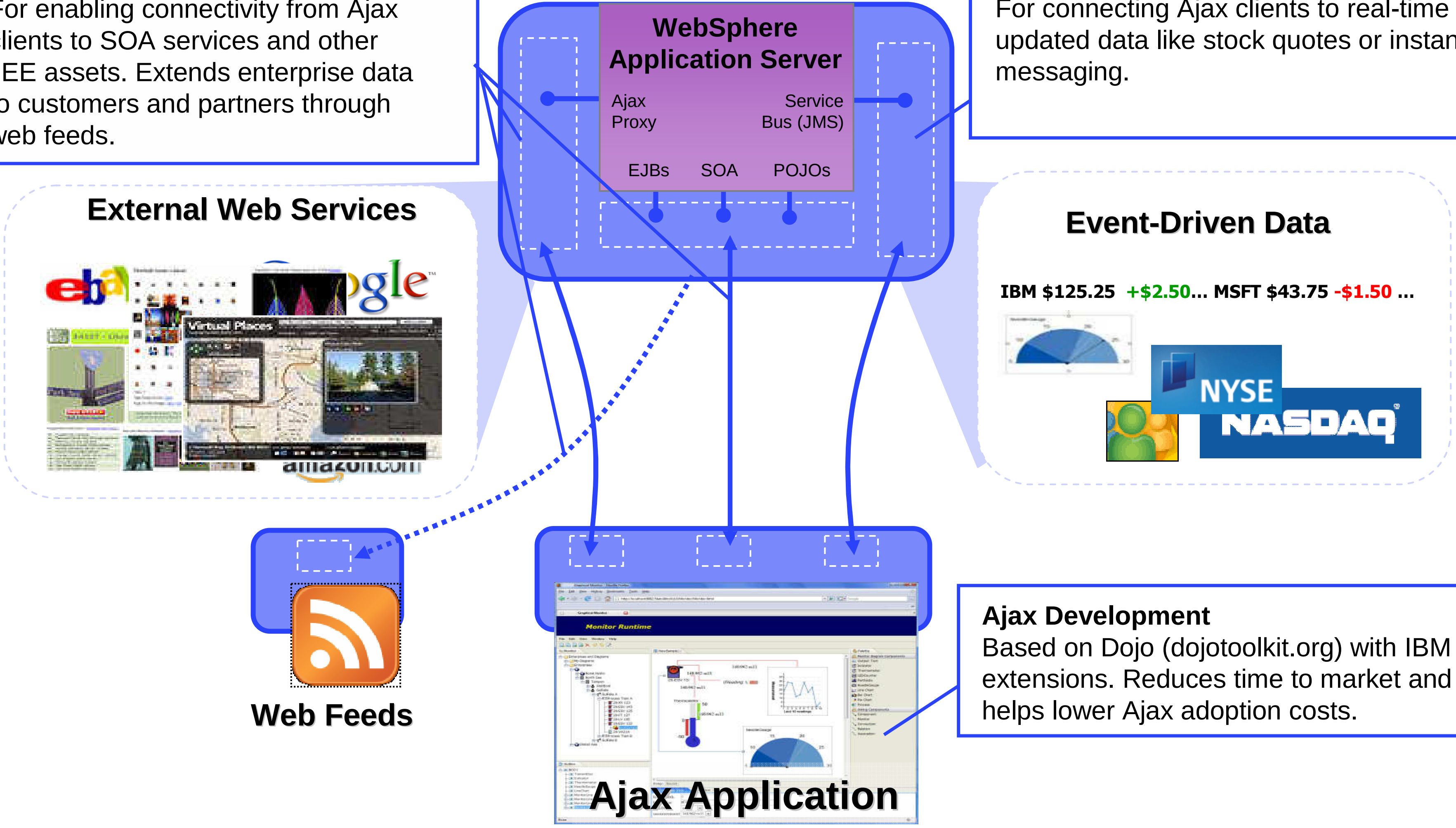
Agenda

- Rational Software Delivery Platform
  - The value RAD adds to your development lifecycle
- Rational Application Developer 7.5
  - EJB 3.0 and JPA
  - Web Tools & JSF Overview
  - Web 2.0 Support
  - Web Services
  - WebSphere support, Server tools, Problem determination
  - WebSphere Portlet and Portal support
  - EIS Adapters
  - Collaboration with other Rational products (RTC, CC plugin…)

WAS Feature Pack for Web 2.0 Highlights

Web 2.0 to SOA Connectivity
For enabling connectivity from Ajax clients to SOA services and other JEE assets. Extends enterprise data to customers and partners through web feeds.

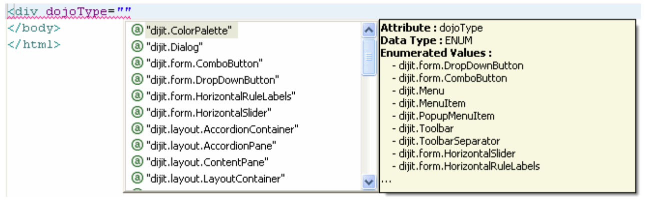
AJAX Messaging
For connecting Ajax clients to real-time updated data like stock quotes or instant messaging.



Ajax Development
Based on Dojo (dojotoolkit.org) with IBM extensions. Reduces time to market and helps lower Ajax adoption costs.

RAD helps create Web 2.0 to SOA Connectivity

- Build AJAX and Dojo clients with rich source level content assist, validation and refactoring



```
<div dojoType=""
</body>
</html>
```

```
"dijit.ColorPalette"
"dijit.Dialog"
"dijit.form.ComboButton"
"dijit.form.DropDownButton"
"dijit.form.HorizontalRuleLabels"
"dijit.form.HorizontalSlider"
"dijit.layout.AccordionContainer"
"dijit.layout.AccordionPane"
"dijit.layout.ContentPane"
"dijit.layout.LayoutContainer"
```

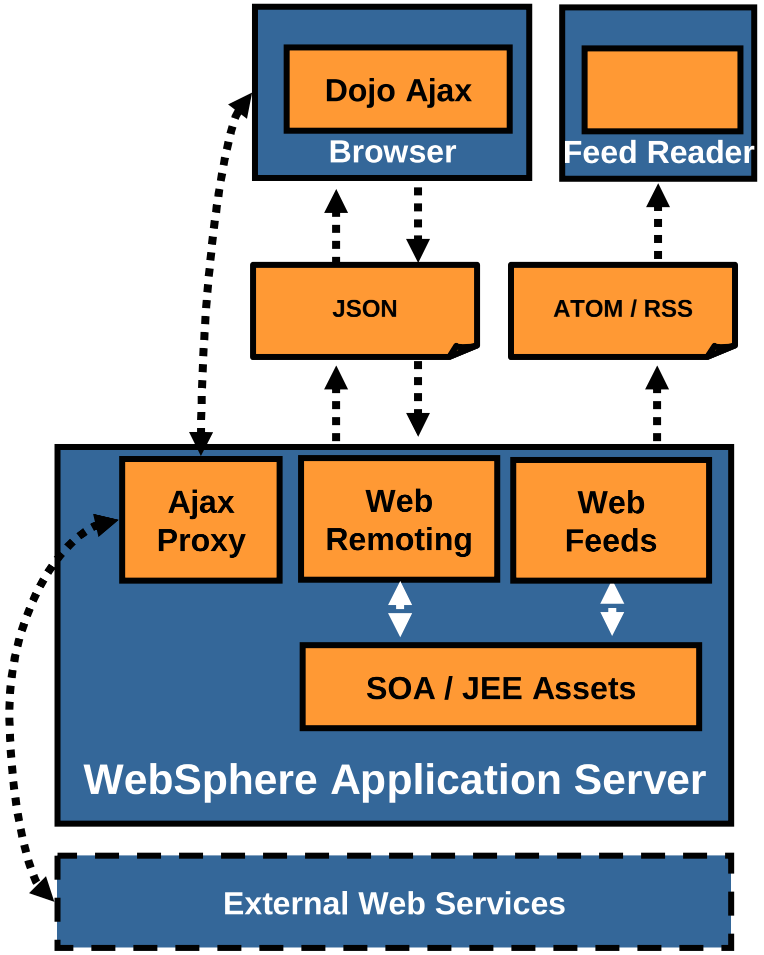
Attribute : dojoType
Data Type : ENUM
Enumerated Values :
- dijit.form.DropDownButton
- dijit.form.ComboButton
- dijit.Menu
- dijit.MenuItem
- dijit.PopupMenuItem
- dijit.Toolbar
- dijit.ToolbarSeparator
- dijit.form.HorizontalSlider
- dijit.form.HorizontalRuleLabels

...

- Visually lay out your client with Page Designer

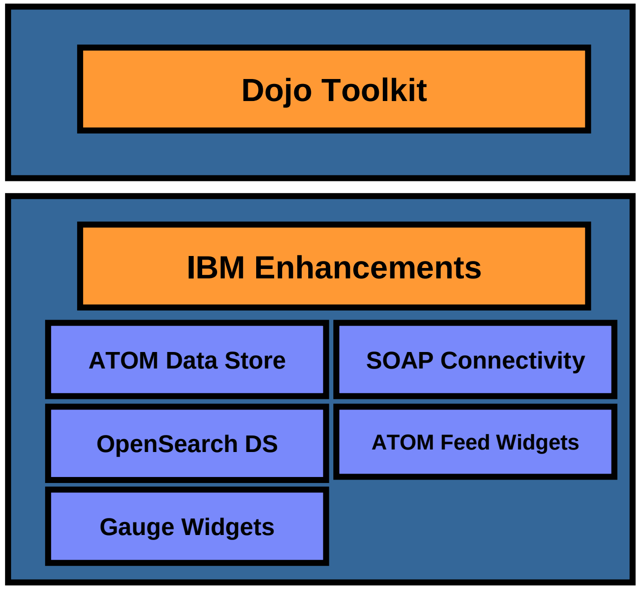
- Wizards to expose server side SOA / Java EE / POJOs with endpoints as REST style services

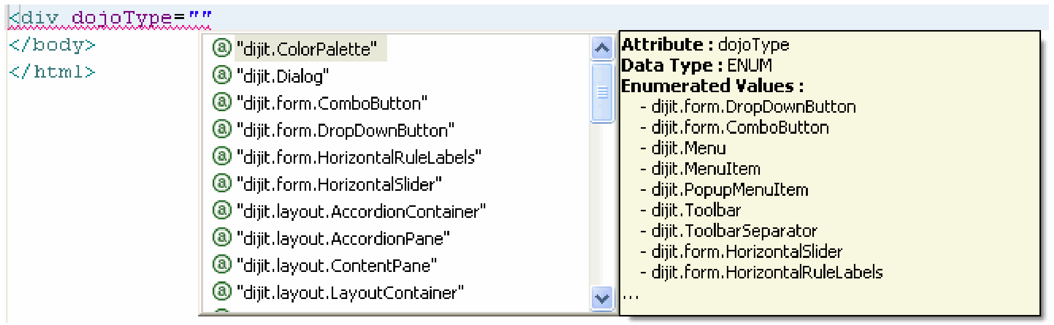
- Javascript debugging and integration with Firebug



RAD helps with RIA client side development

- JavaScript is tricky!
- RAD provides a world class JavaScript source level development environment
  - JavaScript editor with code assist, validation, refactoring, outline view
  - Integration & support for Dojo
    - Dojo specific code assist, validation, refactoring of Dojo tags
    - Based on OpenAJAX IDE Working Group metadata




```
<div dojoType=""
</body>
</html>
```

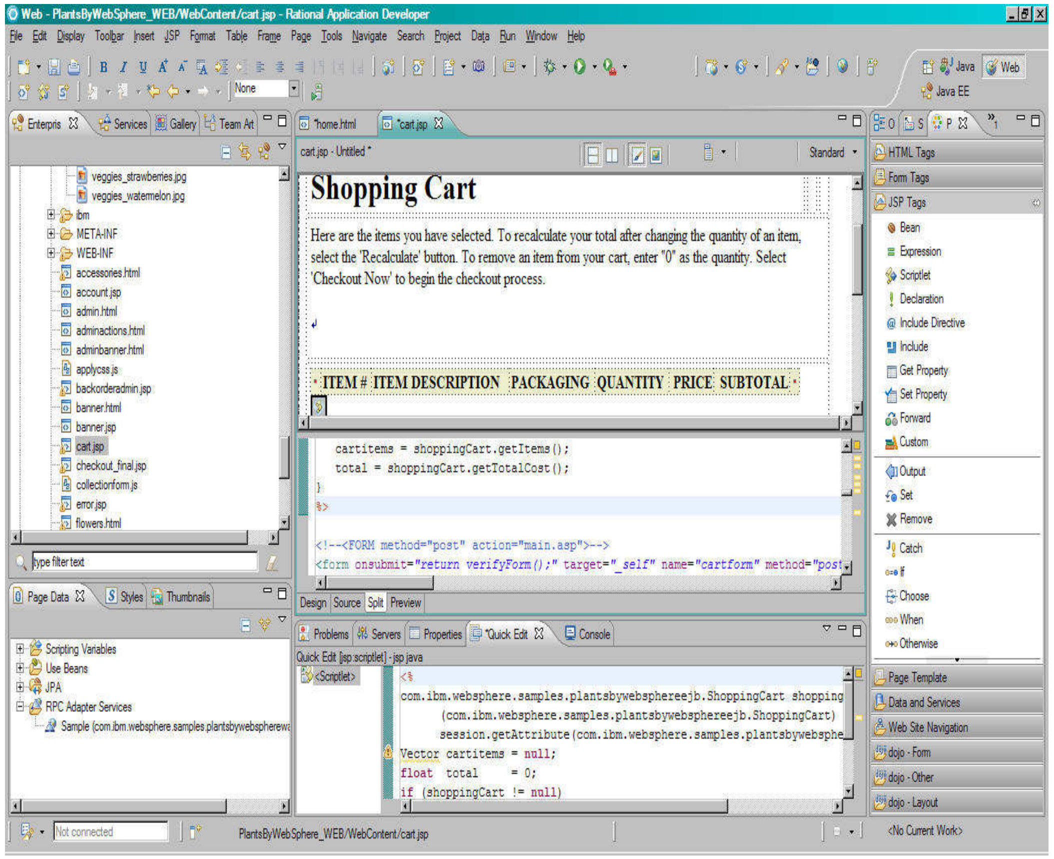
"dijit.ColorPalette"
"dijit.Dialog"
"dijit.form.ComboButton"
"dijit.form.DropDownButton"
"dijit.form.HorizontalRuleLabels"
"dijit.form.HorizontalSlider"
"dijit.layout.AccordionContainer"
"dijit.layout.AccordionPane"
"dijit.layout.ContentPane"
"dijit.layout.LayoutContainer"

Attribute : dojoType
Data Type : ENUM
Enumerated Values :
- dijit.form.DropDownButton
- dijit.form.ComboButton
- dijit.Menu
- dijit.MenuItem
- dijit.PopupMenuItem
- dijit.Toolbar
- dijit.ToolbarSeparator
- dijit.form.HorizontalSlider
- dijit.form.HorizontalRuleLabels

...

RAD helps with RIA Client side Development

- Support for visual construction of RIA pages with Dojo Widgets on Palette for easy drag-and-drop to page
- DOJO property views for setting widget attributes
- Access to REST services, web remoting interfaces and feeds



Shopping Cart

Here are the items you have selected. To recalculate your total after changing the quantity of an item, select the 'Recalculate' button. To remove an item from your cart, enter "0" as the quantity. Select 'Checkout Now' to begin the checkout process.

ITEM #	ITEM DESCRIPTION	PACKAGING	QUANTITY	PRICE	SUBTOTAL

```
    cartitems = shoppingCart.getItems();
    total = shoppingCart.getTotalCost();
}
%>

<!--<FORM method="post" action="main.asp">-->
<form onsubmit="return verifyForm();" target="_self" name="cartform" method="post
```

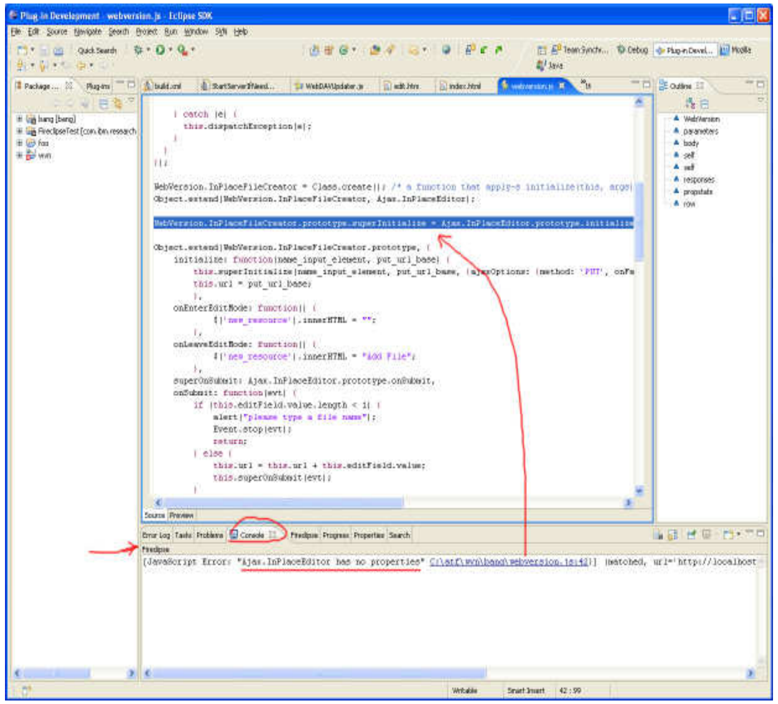
```
<%
com.ibm.websphere.samples.plantsbywebsphereejb.ShoppingCart shopping
        (com.ibm.websphere.samples.plantsbywebsphereejb.ShoppingCart)
        session.getAttribute(com.ibm.websphere.samples.plantsbywebsphe
Vector cartitems = null;
float  total     = 0;
if (shoppingCart != null)
```

RAD helps with RIA server side development

- **Support for exposing methods of Java objects (EJB’s, PoJo’s, web service proxies) via the Web Remoting framework**
  - Endpoints can be used as REST-style Services
- **Support for the Ajax Proxy to allow secure access to internet based services and mashups for external services not in your domain**
- **Server tools support – recognize Web 2.0 feature pack when available**

RAD helps with RIA debugging and testing

- JavaScript debugging
  - Integration with Firebug browser-based debugger
    - AJAX Request Monitor & View
    - DOM inspector
- REST service interaction, JSON data (construct, receive, view) XML data & RSS/Atom Feeds
  - Asynchronous requests



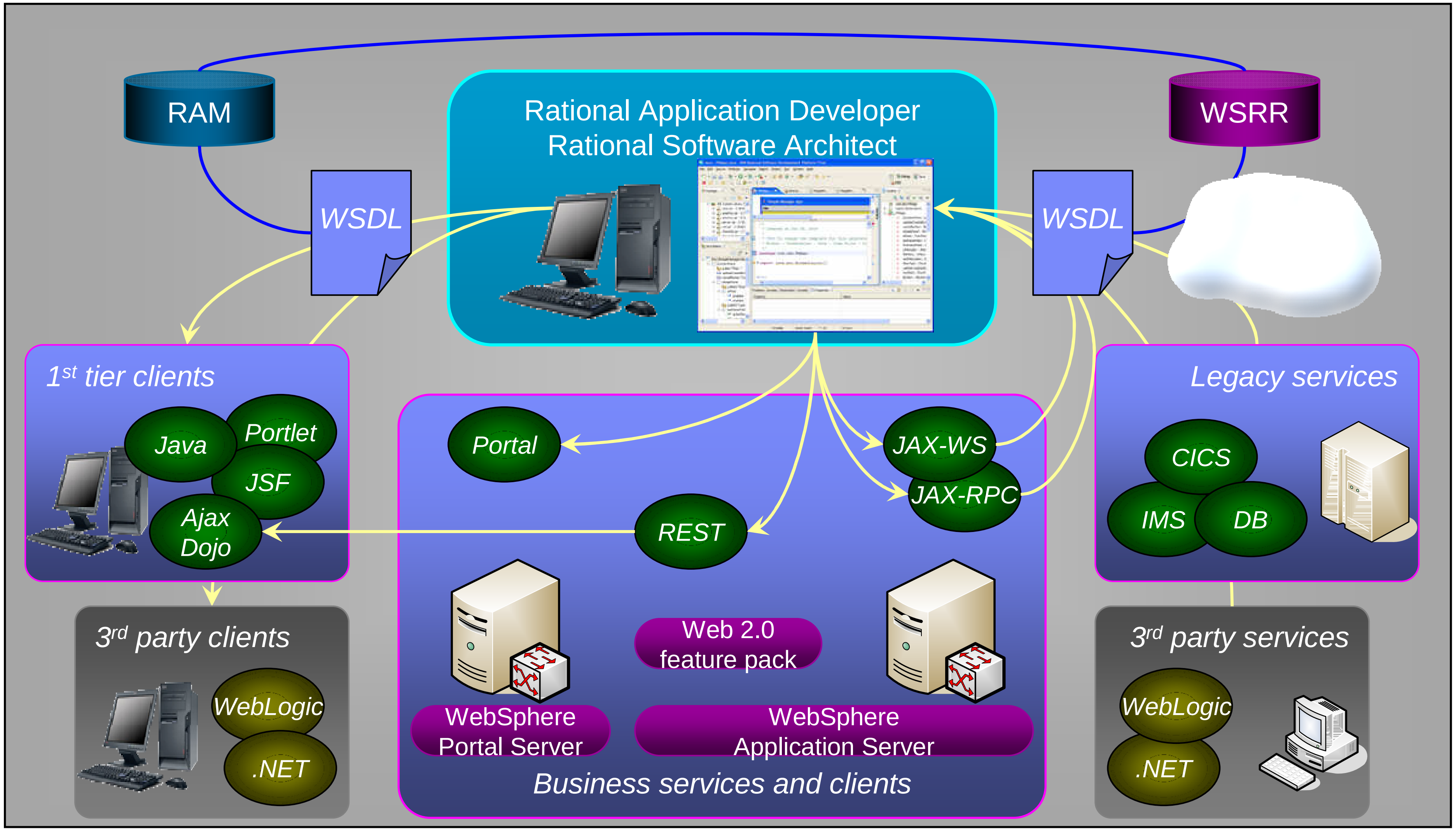
```
        } catch (e) {
          this.dispatchException(e);
        }
      }
});

WebVersion.InPlaceFileCreator = Class.create(); /* a function that apply-s initialize(this, args)
Object.extend(WebVersion.InPlaceFileCreator, Ajax.InPlaceEditor);

WebVersion.InPlaceFileCreator.prototype.superInitialize = Ajax.InPlaceEditor.prototype.initialize

Object.extend(WebVersion.InPlaceFileCreator.prototype, {
    initialize: function(name_input_element, put_url_base) {
        this.superInitialize(name_input_element, put_url_base, {ajaxOptions: {method: 'PUT', onFa
        this.url = put_url_base;
        },
    onEnterEditMode: function() {
            $('new_resource').innerHTML = "";
        },
    onLeaveEditMode: function() {
            $('new_resource').innerHTML = "Add File";
        },
    superOnSubmit: Ajax.InPlaceEditor.prototype.onSubmit,
    onSubmit: function(evt) {
        if (this.editField.value.length < 1) {
            alert("please type a file name");
            Event.stop(evt);
            return;
        } else {
            this.url = this.url + this.editField.value;
            this.superOnSubmit(evt);
        }
```

```
[JavaScript Error: "Ajax.InPlaceEditor has no properties" C:\atf\svn\band\webversion.js:42}] {matched, url='http://localhost
```

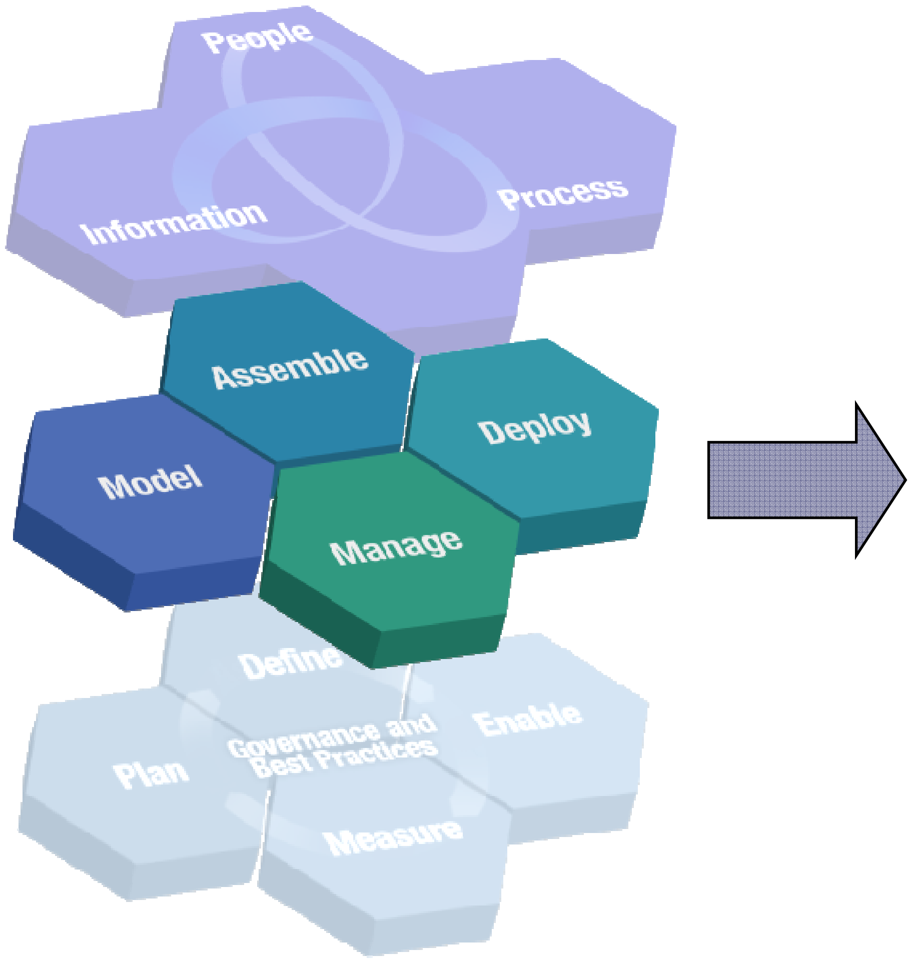
Agenda

- Rational Software Delivery Platform
  - The value RAD adds to your development lifecycle
- RAD v7.5 enhancements
  - JEE5:EJB 3.0 and JPA
  - Web Tools & JSF Overview
  - Web 2.0 Support
  - Web Services
  - WebSphere support, Server tools, Problem determination
  - WebSphere Portlet and Portal support
  - EIS Adapters
  - Collaboration with other Rational products (RTC, CC plugin…)

RAD and RSA for Heterogeneous SOA



Building out IBM’s SOA capabilities



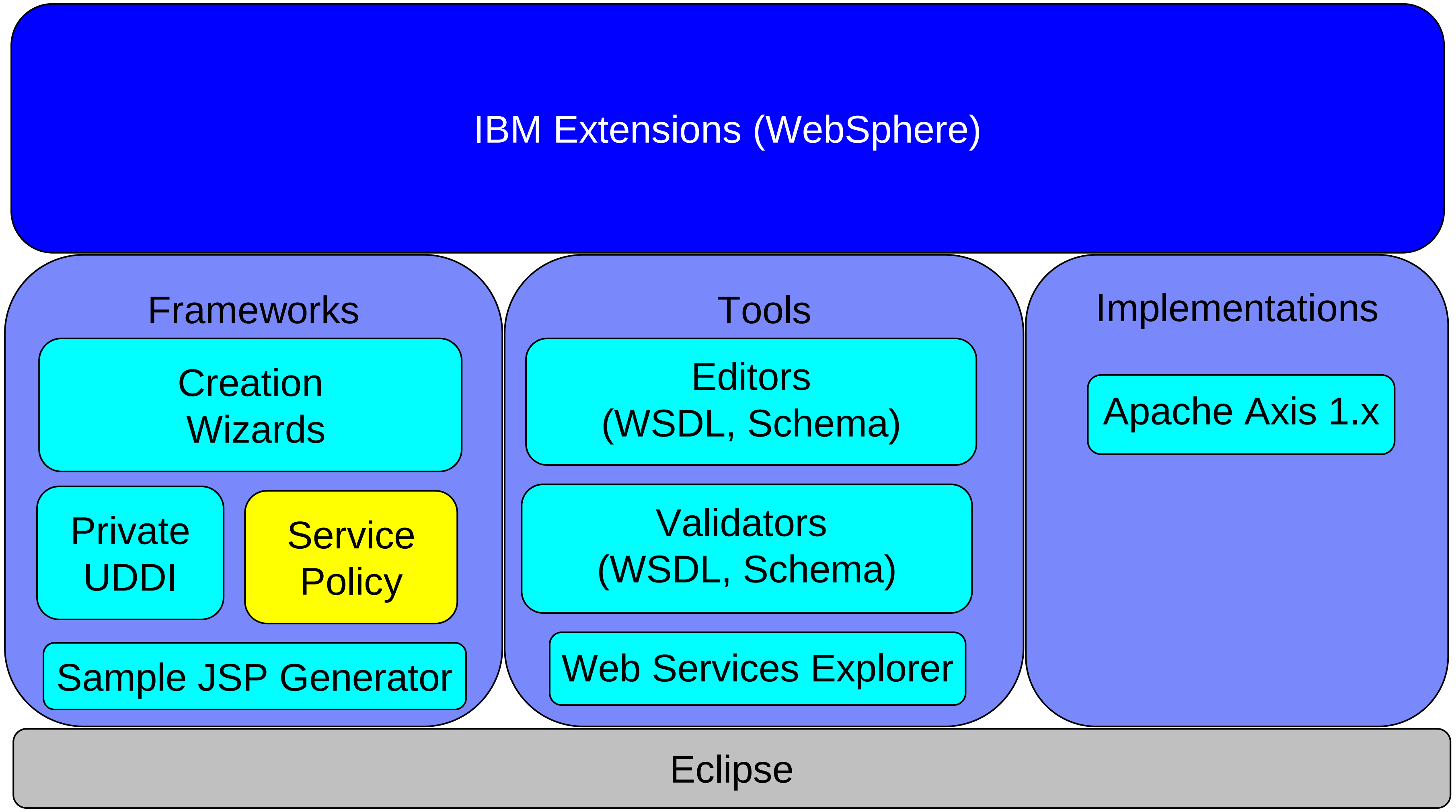
RAD helps realize SOA

- Service creation and reuse
- Service connectivity
- Interaction and collaboration services
- Information as a service

Web Services Development

- What do businesses want next from Web Services?
  - Reliability over HTTP.
  - Asynchronous message exchange.
  - Asynchronous programming model.
  - Conversational security.
  - Binary Data Exchange.
  - Faster XML Parsing.
  - Support for complex XML Schema.
  - A better, and simpler, programming model than JAX-RPC.
- Oh yeah – Keep it Interoperable!

Overview of RAD Web Services Tools



Overview of IBM Extensions for WebSphere

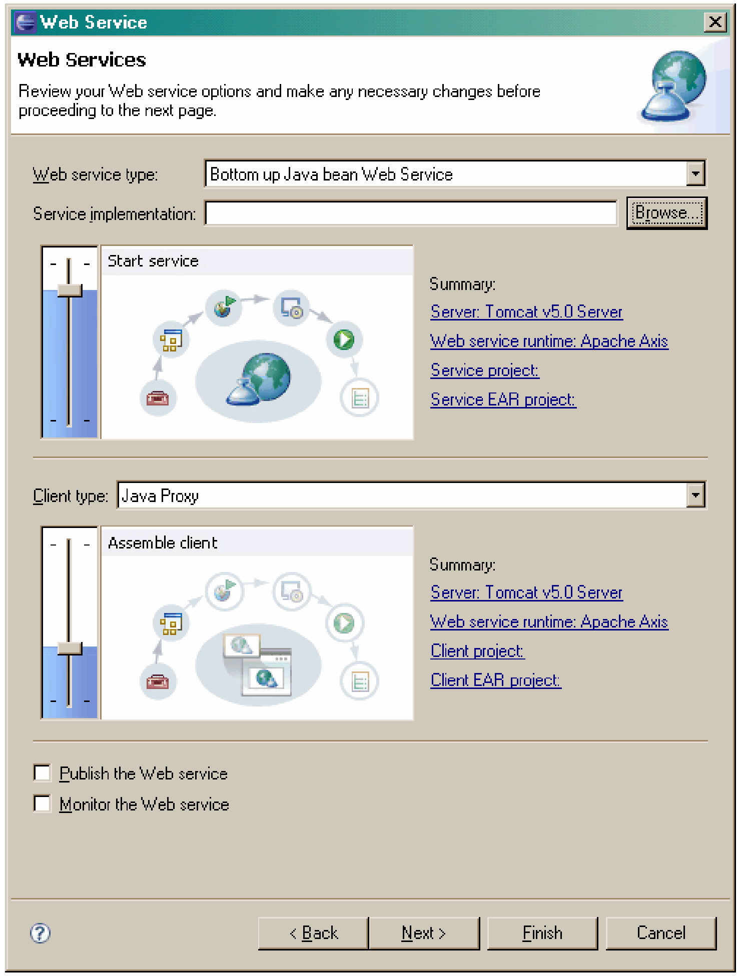
- WebSphere UDDI Registry Configuration Wizard
- Web Service runtimes (Wizard extensions) NEW!
- Support for Editing JAX-WS Annotations
- JAX-WS Annotations Processor
- Quickfixes NEW!
- JAXB Schema to Java Wizard
- Schema Library NEW!
- JSR-109 1.2 Support NEW!
- Service Policy Integration NEW!
- Manage Policy Attachments Wizard NEW!

What RAD supports - Web Services Development

- Java EE 5 Web Services
  - Web Services Annotations, a new Java programming model (JSR-181)
  - JAX-B Schema to Java customizations using JAX-B 2.0 (JSR-222)
  - JAX-WS (Replacement for JAX-RPC) (JSR-224)
  - SOAP 1.2 Bindings in WSDL
- Utilize Sun Reference Implementation for Java EE 5 (JAX-WS and JAX-B)
  - Asynchrony
  - Binary attachment Optimization (MTOM)
- Policy Set simplified/shared configuration for Qualities of Service
  - RM- Reliable Messaging
  - WS-Addressing
  - WS-Security
  - Secure Conversation

Web Services Development

- Discovery of Web services in Page Designer palette
- Simplified WSDL/Schema views in the editor
- Better (performing) validators in WSDL and XML schema validation
- Deployment and testing of Web Services into WebSphere Application server
- Test Web Service client with Universal Test client


Web Service

Web Services

Review your Web service options and make any necessary changes before proceeding to the next page.

Web service type: Bottom up Java bean Web Service

Service implementation:

Start service

Summary:
Server: Tomcat v5.0 Server
Web service runtime: Apache Axis
Service project:
Service EAR project:

Client type: Java Proxy

Assemble client

Summary:
Server: Tomcat v5.0 Server
Web service runtime: Apache Axis
Client project:
Client EAR project:

☐ Publish the Web service
☐ Monitor the Web service

Web Service Metadata Annotations (JSR 181)

Sample JAX-WS Web Service:

```
import javax.jws.WebService;
import javax.jws.WebMethod;

@WebService
public class Echo {
  @WebMethod
  public String echoString(String input)  {
    return input;
  }

  public int echoInt(int input)  {
    return input;
  }
}
```

Only echoString() will appear as a WSDL operation.

JAX-B Schema to Java Bean Generation (JSR 222)

- JAXB 2.0 provides full support of all XML Schema features, significantly fewer generated classes, generated classes that are easier to manipulate, and a more flexible validation mechanism

- RAD wizard that takes your schema file and generates JAXB classes

- RAD wizard includes schema library support

  - If selected, by default, each schema will have a project created for it or user can change this to whatever project grouping makes sense for their usage pattern

Web Services: Quality of Service support

Quality of service (QOS):

- The ability to provide different priority to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow

Policy Sets:

- Use policy sets to simplify configuring the qualities of service for Web services and clients.
- Policy sets are assertions about how Web Services are defined.
- Using policy sets, you can combine policy types.
- Policy Sets can be defined at the global level or on a specific project

Policy Sets were first introduced to the WebSphere Application Server in the Web Services Feature Pack

Web Services: RAD support for QOS

- Developers can use the WebSphere Admin console to create custom Policy Sets and then import them to RAD (or vice versa)
- Developers can modify the custom binding associated with a policy type within RAD.
  - RAD will validate any changes made to the custom binding files

Example of Policy Sets and Policy Types within them:

- **WS-Security default:**
  - WS-Security, WS-Addressing
- **Reliable Asynchronous Messaging Profile (RAMP) default:**
  - WS-Security, WS-Addressing, WS-Reliable Messaging.
- **WS HTTPS default:**
  - HTTP Transport, SSL Transport, WS-Addressing
- **Reliable Messaging:**
  - WS-Reliable Messaging.

Agenda

- Rational Software Delivery Platform
  - The value RAD adds to your development lifecycle
- RAD v7.5 enhancements
  - JEE5:EJB 3.0 and JPA
  - Web Tools & JSF Overview
  - Web 2.0 Support
  - Web Services
  - WebSphere support, Server tools, Problem determination
  - WebSphere Portlet and Portal support
  - EIS Adapters
  - Collaboration with other Rational products (RTC, CC plugin…)

WebSphere Test Environment support

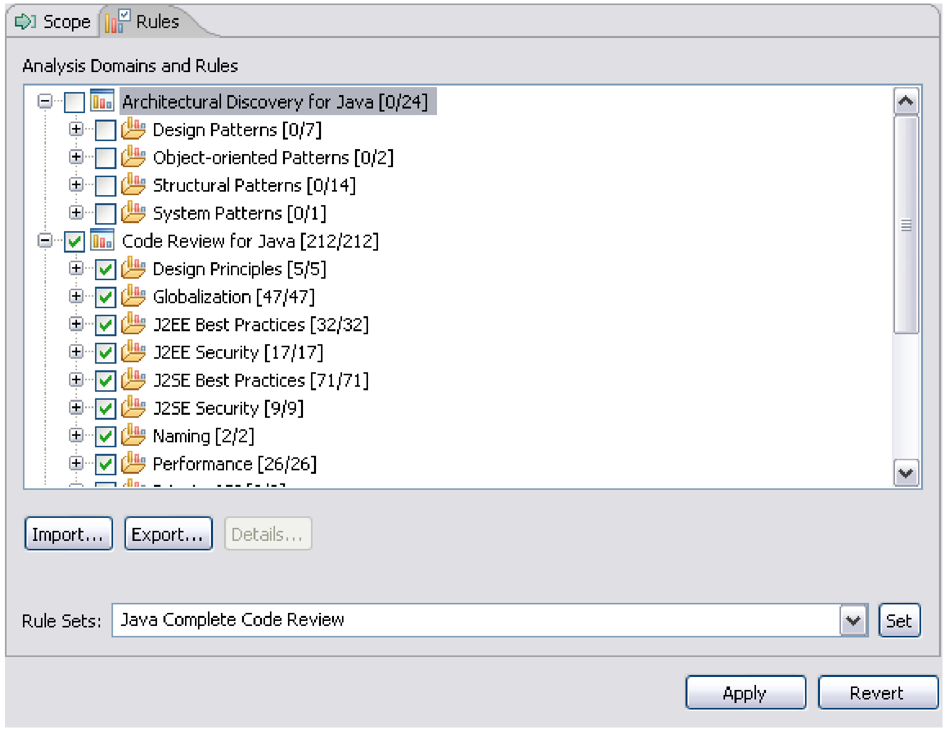
- **WebSphere Application Server 6.0**
  - Includes support for Web 2.0 feature pack
- **WebSphere Application Server 6.1**
  - Includes support for EJB 3.0, Web Services and Web 2.0 feature pack
- **WebSphere Application Server 7.0**
  - Includes support for Web 2.0 feature pack
- Remote deployment to all WebSphere platforms above
- Incremental publish
- Integrated debugging, menu items for admin console, ability launch WSADMIN, application client launcher
- Universal test client to dynamically test your applications

RAD helps with improving Quality of applications

- **Line Level Code Coverage**
  - Code coverage of class/method/block/line for package/class/methods
  - Filtering to include/exclude packages, classes, methods
  - Eclipse (Java editor integration) and HTML (portable/BIRT) reports.
  - Enables code coverage from Ant
  - Support for generating code coverage statistics for web applications
- **Application profiling**
  - Running an application in profiling mode allows the performance of the application to be traced and improve
  - Aids in program understanding by showing execution patterns

Code Quality Assurance

- Analyze Project/Workspace to find problems of various types:
  - Design Principles
  - Globalization
  - J2EE & J2SE Best Practices
  - J2EE & J2SE Security
  - Naming
  - Performance
  - Private API
- Produce interactive reports with violations and metrics
- Provides explanations, examples, and quick fixes for problems
- Allow users to create, enable and disable validation rules
- Allow users to create their own rules based on rule templates
  - Complete Code Review (200+ rules)



Analysis Domains and Rules

- [ ] Architectural Discovery for Java [0/24]
  - [ ] Design Patterns [0/7]
  - [ ] Object-oriented Patterns [0/2]
  - [ ] Structural Patterns [0/14]
  - [ ] System Patterns [0/1]
- [✓] Code Review for Java [212/212]
  - [✓] Design Principles [5/5]
  - [✓] Globalization [47/47]
  - [✓] J2EE Best Practices [32/32]
  - [✓] J2EE Security [17/17]
  - [✓] J2SE Best Practices [71/71]
  - [✓] J2SE Security [9/9]
  - [✓] Naming [2/2]
  - [✓] Performance [26/26]

Rule Sets: Java Complete Code Review

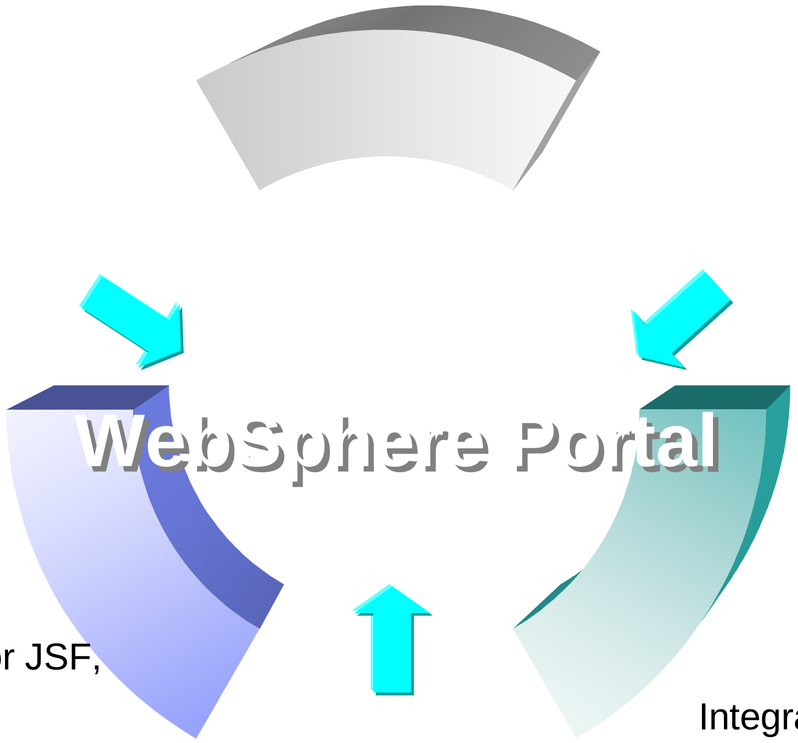
Debug Tools

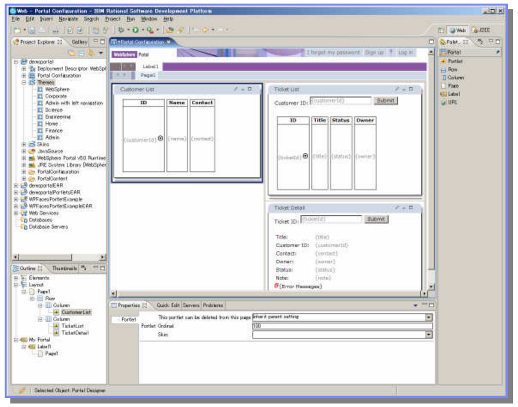
- Java and mixed language debugger
  - Seamless integration when debugging application that calls other languages from Java and vice versa
- J2EE/Web application debugging
  - Advanced debug support for Websphere Application Server, including EJBs, JSP pages, and servlets.
- Debugger for Jython based WebSphere Administration Scripts
- Support for DB2 V9 Stored Procedure Debug
  - Java and SQL Stored procedures
- Step-by-Step Debugging
  - Control debugging at a higher level. Provide user the ability to stop on entry to every object loaded by the JVM or server.
- XSLT debugger
  - Allows users to detect and diagnose errors in XSLT Transformations
- Logical display of complex variable types
  - Display variables in a logical manner, allowing the user to examine variables more easily.

Agenda

- Rational Software Delivery Platform
  - The value RAD adds to your development lifecycle
- RAD v7.5 enhancements
  - JEE5:EJB 3.0 and JPA
  - Web Tools & JSF Overview
  - Web 2.0 Support
  - Web Services
  - WebSphere support, Server tools, Problem determination
  - WebSphere Portlet and Portal support
  - EIS Adapters
  - Collaboration with other Rational products (RTC, CC plugin…)

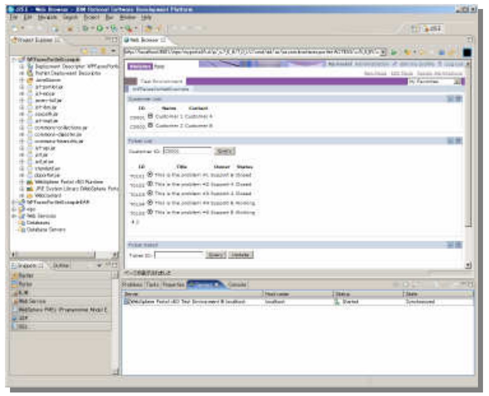
1st Class Support for Portal App Development





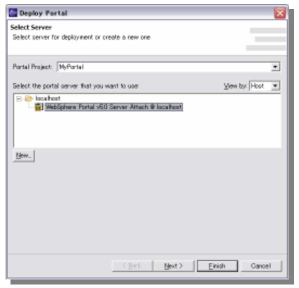
Visual Portlet & Portal Site Development

- ✓ Integrated Portlet support for JSF, Struts framework
- ✓ Portlet templates
- ✓ Portal page Layout
- ✓ Editing of Themes and Skins
- ✓ IBM Portlet API and JSR 168 Portlet API support



Integrated Portal Test Environment
- ✓ WebSphere Portal 6.1

Integrated WebSphere Test Environment for Portlet Applications
WebSphere Portal 6.0 (stub support)



Deploy Portal

Select Server
Select server for deployment or create a new one

Portal Project: MyPortal

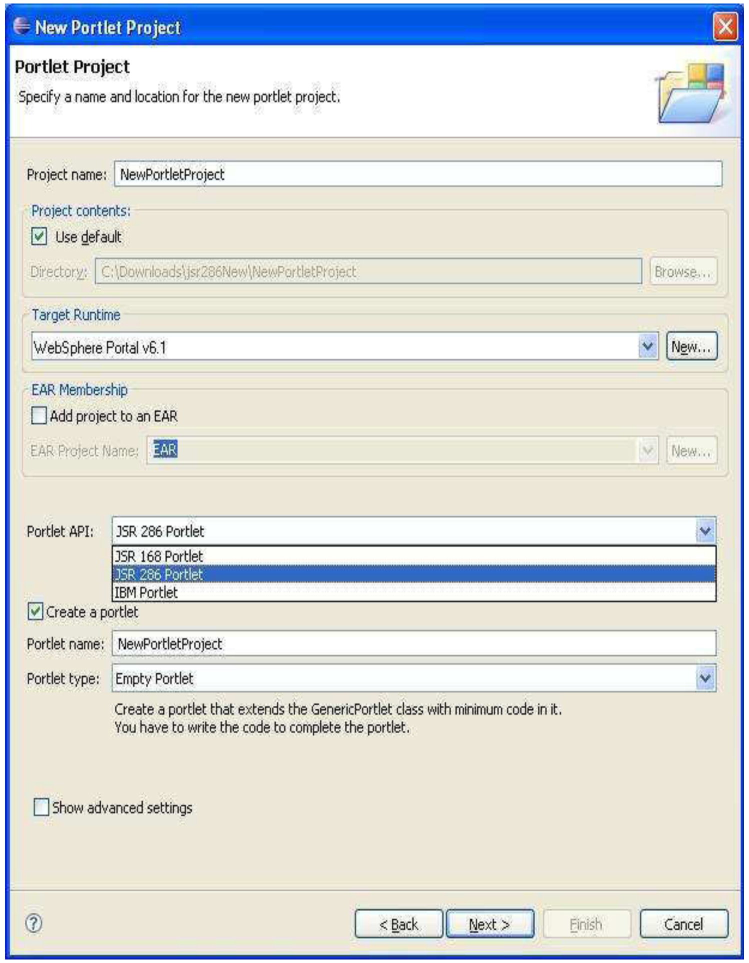
Select the portal server that you want to use — View by: Host

localhost
WebSphere Portal v6.0 Server Attach @ localhost

Import & Deploy Wizard

JSR 286 Support

- Creation of JSR 286 Portlet Project
- Support for Portlet Events:
  - JSR 286 allows the Portlets to declare events it wants to publish (send), and events it wants to process (receive).
- Support for Resource Serving: JSR 286 allows Portlets to serve resources.
  - The resources may be images, jsps and so on. The Portlet can serve resource using resource URLs.
  - The Portlet tooling will address the code generation done as a result of adding <portlet:resourceURL> tag in the Portlet JSP.


New Portlet Project

Portlet Project

Specify a name and location for the new portlet project.

Project name: NewPortletProject

Project contents:

☑ Use default

Directory: C:\Downloads\jsr286New\NewPortletProject

Target Runtime: WebSphere Portal v6.1

EAR Membership

☐ Add project to an EAR

EAR Project Name: EAR

Portlet API: JSR 286 Portlet (JSR 168 Portlet, JSR 286 Portlet, IBM Portlet)

☑ Create a portlet

Portlet name: NewPortletProject

Portlet type: Empty Portlet

Create a portlet that extends the GenericPortlet class with minimum code in it.
You have to write the code to complete the portlet.

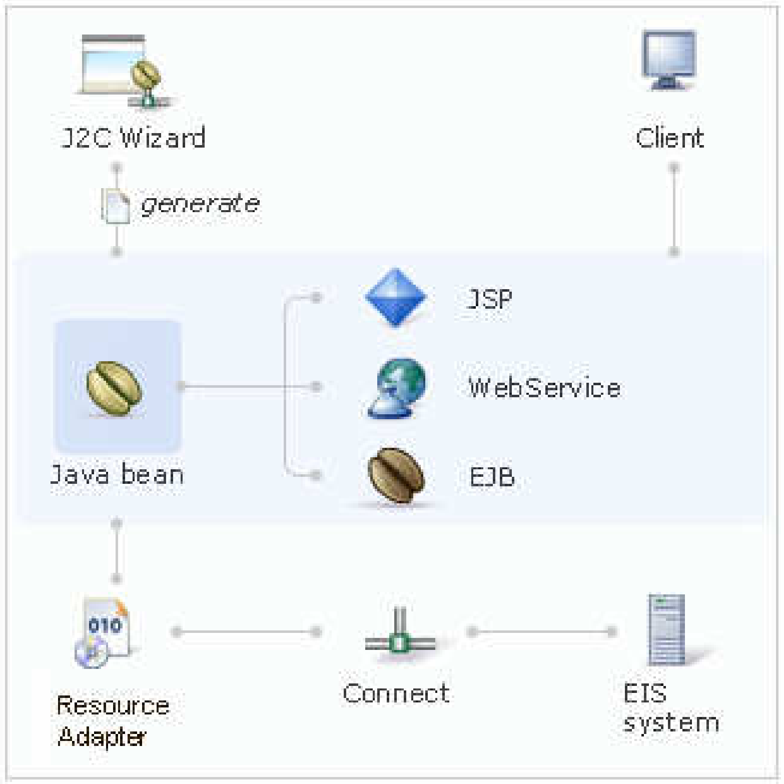
☐ Show advanced settings

Agenda

- Rational Software Delivery Platform
    - The value RAD adds to your development lifecycle
- RAD v7.5 enhancements
    - JEE5:EJB 3.0 and JPA
    - Web Tools & JSF Overview
    - Web 2.0 Support
    - Web Services
    - WebSphere support, Server tools, Problem determination
    - WebSphere Portlet and Portal support
    - EIS Adapters
    - Collaboration with other Rational products (RTC, CC plugin…)

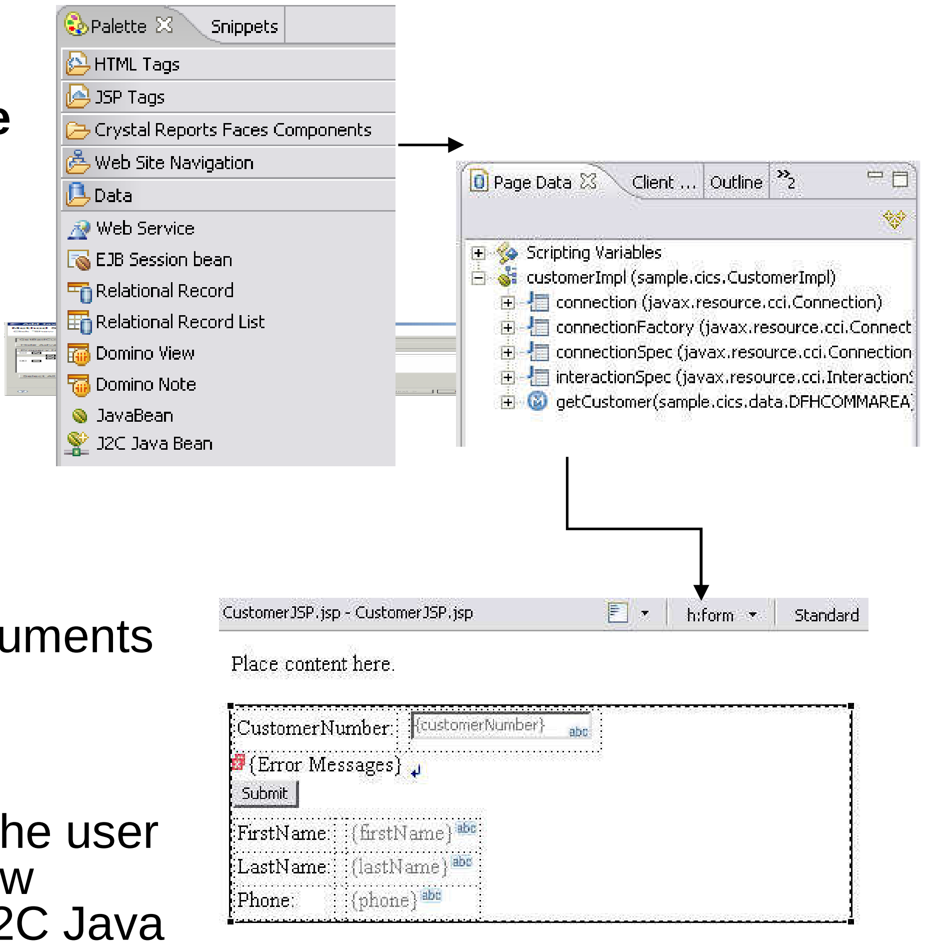
What's new - WebSphere Application Adapters

- Tools supporting development time adapters for:
  - SAP
  - PeopleSoft Enterprise
  - Siebel
  - Oracle E-Business Suite
  - JD Edwards
- Outbound support for WAS Adapters
  - J2C Java bean wizard
    - Live connection to discover objects and methods
- Edit data type's schema
  - Refactoring and regeneration of Java data binding and J2C bean
- Support deploy options for new adapters in the existing deploy wizard
  - Simple JSP, Faces JSP, EJB, Web Services



J2C Tools

- **Support for CICS ECI and IMS resource adapters**
- **CICS Transaction Gateway included (developer use)**
- **Page Designer integration**
  - Palette entry for J2C Java beans for drag and drop
- **Enhanced Editing Support**
  - Wizard based guidance to expose J2C InteractionSpec properties as input arguments
    - E.g.. User name, password
- **Wizard Session Recording**
  - Creates an Ant build file that captures the user interaction with the J2C Wizards to allow command line based regeneration of J2C Java Beans and the Language Data Beans
- **COBOL, C, MFS and PL/I as supported native languages**



Agenda

- Rational Software Delivery Platform
    - The value RAD adds to your development lifecycle
- RAD v7.5 enhancements
    - JEE5:EJB 3.0 and JPA
    - Web Tools & JSF Overview
    - Web 2.0 Support
    - Web Services
    - WebSphere support, Server tools, Problem determination
    - WebSphere Portlet and Portal support
    - EIS Adapters
    - Collaboration with other Rational products (RTC, CC plugin…)

Introducing IBM Rational Team Concert

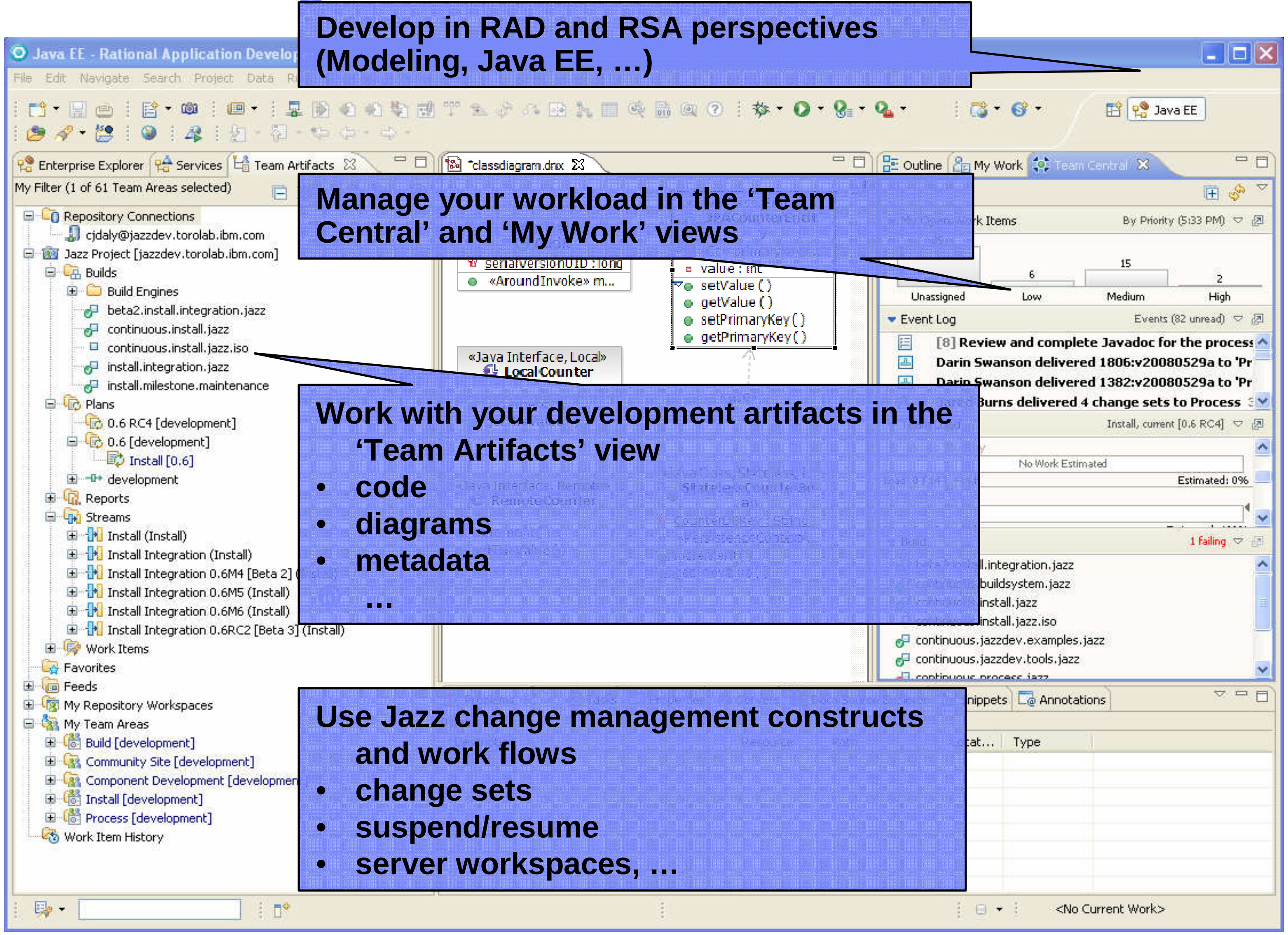
Software innovation through collaboration

- Enables “real-time, in-context” collaboration for distributed project teams – making software development more automated, transparent and predictive
- Integrates source control, work item, reporting and build capabilities which “think and work in unison”
- Provides real-time project health information and transparency of status through automated data gathering
- Allows choice of client tools and extends the value of ClearQuest & ClearCase in enterprise deployments

Open and extensible on Jazz

✓ Collaborate in context
✓ Right-size governance
✓ Day one productivity

RTC Views Integrated into RAD / RSA



Develop in RAD and RSA perspectives (Modeling, Java EE, …)

Manage your workload in the 'Team Central' and 'My Work' views

Work with your development artifacts in the 'Team Artifacts' view
- **code**
- **diagrams**
- **metadata**
- **…**

Use Jazz change management constructs and work flows
- **change sets**
- **suspend/resume**
- **server workspaces, …**

Java EE - Rational Application Develop

File Edit Navigate Search Project Data

Enterprise Explorer | Services | Team Artifacts

My Filter (1 of 61 Team Areas selected)

- Repository Connections
  - cjdaly@jazzdev.torolab.ibm.com
- Jazz Project [jazzdev.torolab.ibm.com]
  - Builds
    - Build Engines
    - beta2.install.integration.jazz
    - continuous.install.jazz
    - continuous.install.jazz.iso
    - install.integration.jazz
    - install.milestone.maintenance
  - Plans
    - 0.6 RC4 [development]
    - 0.6 [development]
      - Install [0.6]
    - development
  - Reports
  - Streams
    - Install (Install)
    - Install Integration (Install)
    - Install Integration 0.6M4 [Beta 2] (Install)
    - Install Integration 0.6M5 (Install)
    - Install Integration 0.6M6 (Install)
    - Install Integration 0.6RC2 [Beta 3] (Install)
  - Work Items
- Favorites
- Feeds
- My Repository Workspaces
- My Team Areas
  - Build [development]
  - Community Site [development]
  - Component Development [development]
  - Install [development]
  - Process [development]
- Work Item History

*classdiagram.dnx

serialVersionUID : long
«AroundInvoke» m...
value : int
setValue ( )
getValue ( )
setPrimaryKey ( )
getPrimaryKey ( )
«Java Interface, Local»
LocalCounter

Outline | My Work | Team Central

My Open Work Items — By Priority (5:33 PM)

Unassigned	Low	Medium	High
	6	15	2

Event Log — Events (82 unread)

[8] Review and complete Javadoc for the process
Darin Swanson delivered 1806:v20080529a to 'Pr
Darin Swanson delivered 1382:v20080529a to 'Pr
Jared Burns delivered 4 change sets to Process

Install, current [0.6 RC4]

No Work Estimated

Estimated: 0%

Build — 1 failing

beta2.install.integration.jazz
continuous.buildsystem.jazz
continuous.install.jazz
continuous.install.jazz.iso
continuous.jazzdev.examples.jazz
continuous.jazzdev.tools.jazz
continuous.process.jazz

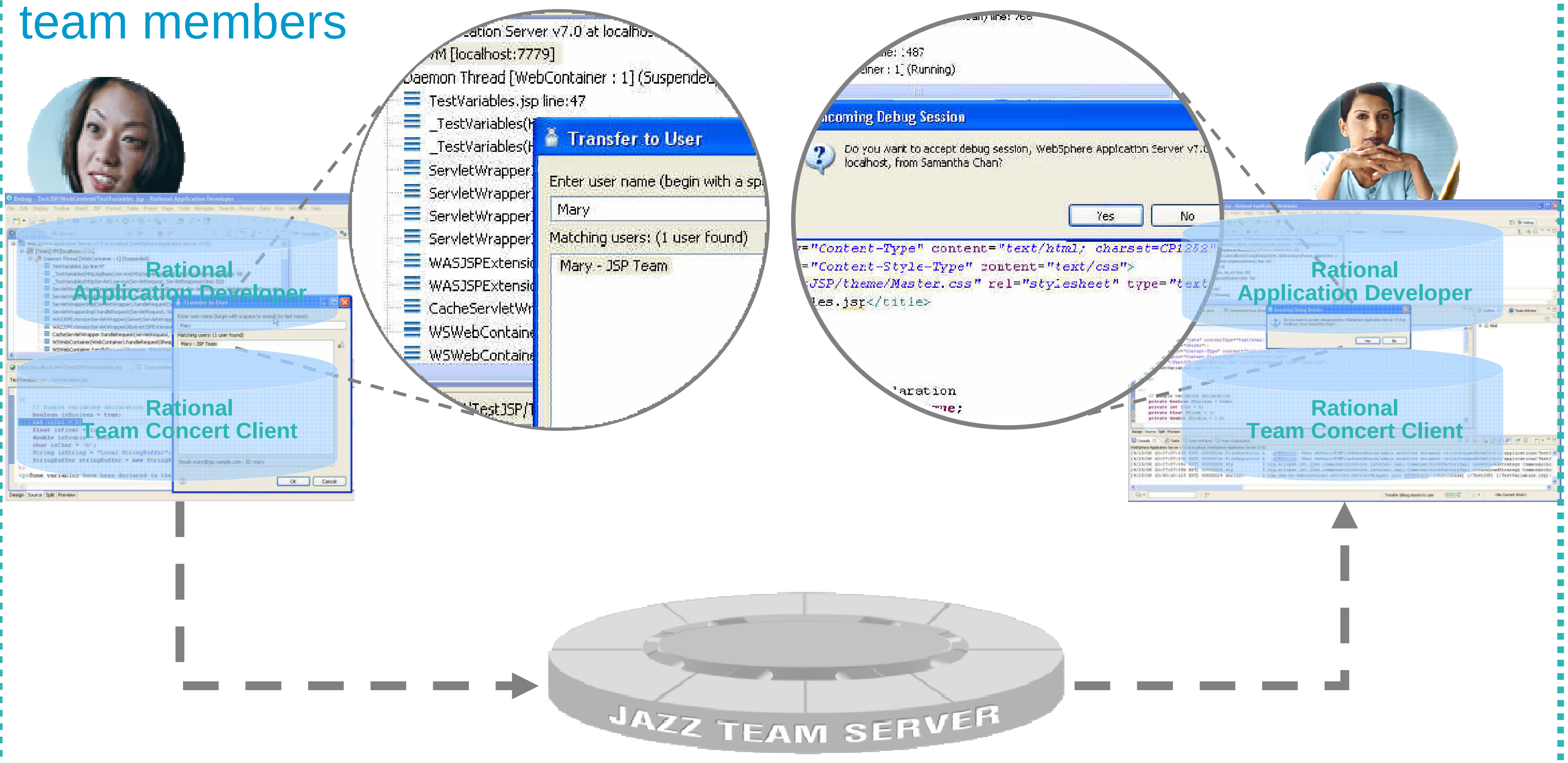
Snippets | Annotations

Location | Type

<No Current Work>

Leveraging Rational Team Concert for Collaborative Debugging

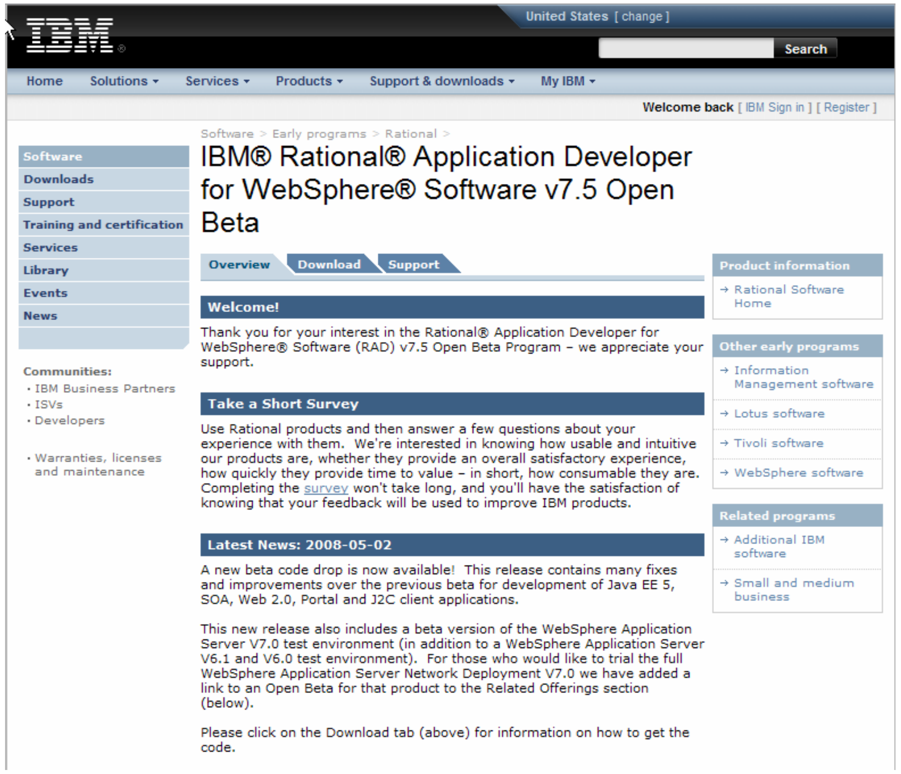
With RAD/RSA and RTC you can share live debug sessions between team members



```
ation Server v7.0 at localho
VM [localhost:7779]
Daemon Thread [WebContainer : 1] (Suspended
TestVariables.jsp line:47
_TestVariables(H
_TestVariables(H
ServletWrapper
ServletWrapper
ServletWrapper
ServletWrapper
WASJSPExtensio
WASJSPExtensio
CacheServletWr
WSWebContaine
WSWebContaine
```

```
me: :487
ainer : 1] (Running)
="Content-Type" content="text/html; charset=CP1252"
="Content-Style-Type" content="text/css">
JSP/theme/Master.css" rel="stylesheet" type="text
les.jsp</title>
laration
```

For More Information on the RAD v7.5 Open Beta



Welcome back [IBM Sign in] [Register]

Software > Early programs > Rational >

IBM® Rational® Application Developer for WebSphere® Software v7.5 Open Beta

Welcome!

Thank you for your interest in the Rational® Application Developer for WebSphere® Software (RAD) v7.5 Open Beta Program – we appreciate your support.

Take a Short Survey

Use Rational products and then answer a few questions about your experience with them. We're interested in knowing how usable and intuitive our products are, whether they provide an overall satisfactory experience, how quickly they provide time to value – in short, how consumable they are. Completing the survey won't take long, and you'll have the satisfaction of knowing that your feedback will be used to improve IBM products.

Latest News: 2008-05-02

A new beta code drop is now available! This release contains many fixes and improvements over the previous beta for development of Java EE 5, SOA, Web 2.0, Portal and J2C client applications.

This new release also includes a beta version of the WebSphere Application Server V7.0 test environment (in addition to a WebSphere Application Server V6.1 and V6.0 test environment). For those who would like to trial the full WebSphere Application Server Network Deployment V7.0 we have added a link to an Open Beta for that product to the Related Offerings section (below).

Please click on the Download tab (above) for information on how to get the code.

Communities:
- IBM Business Partners
- ISVs
- Developers

- Warranties, licenses and maintenance

Product information
- → Rational Software Home

Other early programs
- → Information Management software
- → Lotus software
- → Tivoli software
- → WebSphere software

Related programs
- → Additional IBM software
- → Small and medium business

Technical Resources on IBM developerWorks
- www.ibm.com/developerworks/rational
- Technical library of whitepapers, utilities, betas
- Downloadable demos
- Discussion forums

Rational Application Developer 7.5 Open Beta:

- https://www14.software.ibm.com/iwm/web/cc/earlyprograms/rational/RAD75OpenBeta/

Thank You

BACKUP

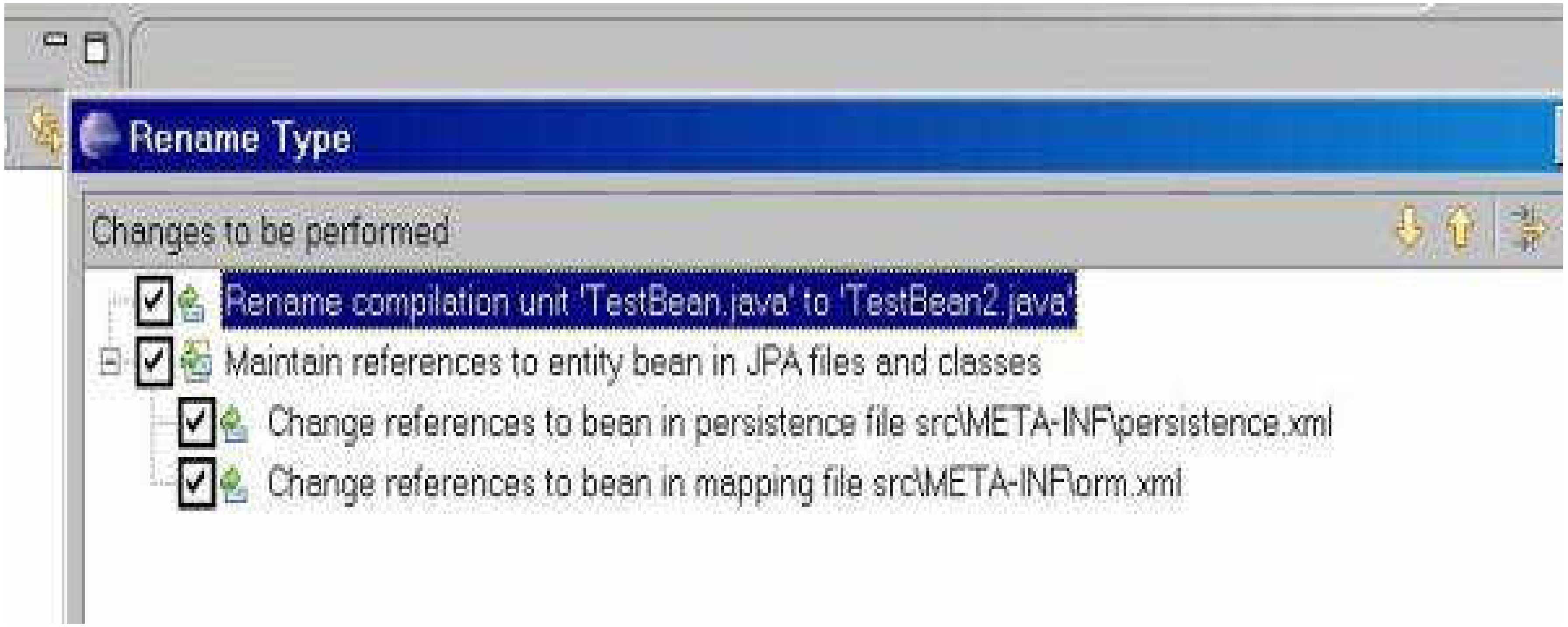
Extra Java EE Tools

RAD helps with - EE 5 Refactoring Operations

- EJB3
  - Extract business interface
  - Promote method to business interface
  - Rename/move EJB session class
  - Rename business interface
  - Rename dependency injection
  - Rename interceptor class
- JPA
  - Rename/move JPA class

RAD helps with - EJB 3.0 and JPA Refactoring

Rename/Move JPA and EJB 3 classes

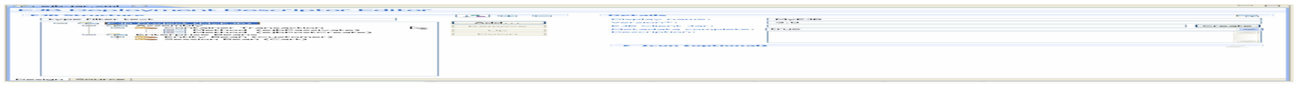


- [✓] Rename compilation unit 'TestBean.java' to 'TestBean2.java'
- [✓] Maintain references to entity bean in JPA files and classes
  - [✓] Change references to bean in persistence file src\META-INF\persistence.xml
  - [✓] Change references to bean in mapping file src\META-INF\orm.xml

RAD helps with - EJB 3.0/EAR Deployment Descriptor Editors

Common look and feel between all XML deployment descriptor editors

- Most values defaulted
- Indicators for required fields



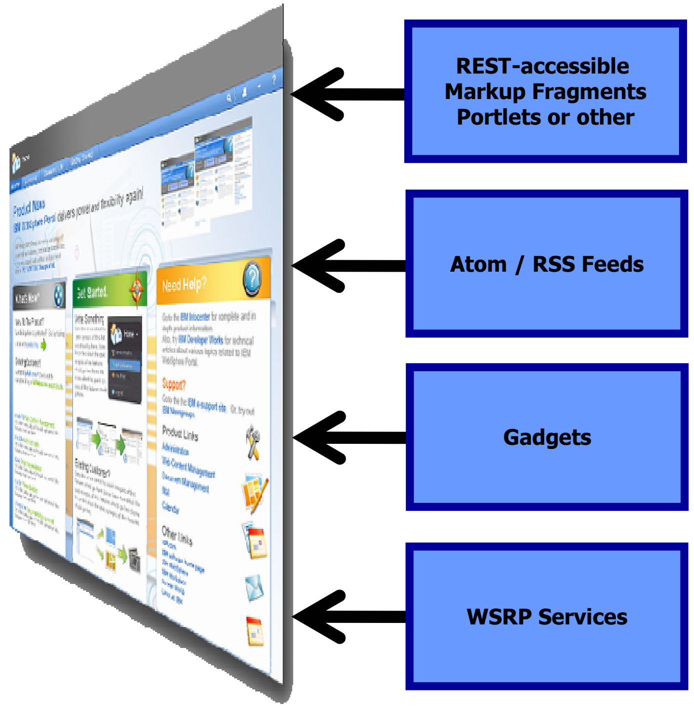
Extra Web Tools

Web Tools in RAD V7.5

- Site Designer
  - Create, manage and build your website
- Web Diagram Editor for Model – View – Controller design
  - Visual application flows
  - Supports Struts and JSF applications
- Page Designer
  - Visually layout your pages using the web technology of your choice:
    - Static HTML
    - Struts support (1.2 and 1.3)
    - JSF 1.2 runtime support
      - Visual development of JSF-based pages using Page Designer
        - Built-in Component Property Editor
        - Built-in tools simplify/automate event handling
        - Built-in tools simplify page navigation
    - SDO support
  - JSF based report viewing for embedding reports in web applications
  - Web development templates and samples

Extra Portal content

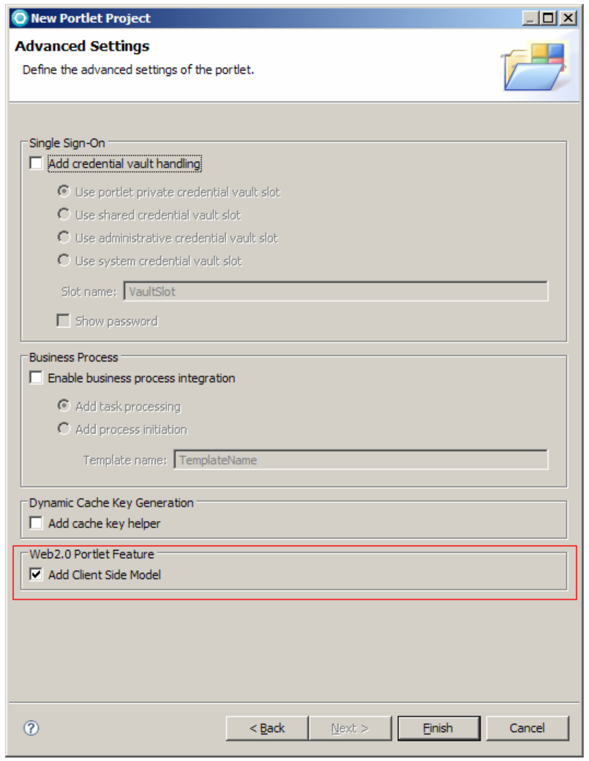
Web 2.0 Client Side Aggregation



- Browser-side Aggregation, Navigation and Customization
  - Renders XML obtained from the server on the browser side
  - Implemented using AJAX, XML, Dojo, and JavaScript
  - Accesses and manipulates Portal through REST* services
- Superior user experience
  - Highly reactive and direct user interface
  - Many actions possible without server roundtrips
  - Avoids page flickering
- Improved performance and scalability through
  - Reduced server side processing - offloads rendering to browser
  - Reduced bandwidth requirements between server and browser
  - Reduced client-side processing – mostly fragment reloads, few page reloads
  - Improved caching, all artifacts can be cached independently

Support for Client Side Programming Model

- One click to enable Web 2.0 functionality
- Improves performance
- Reduces repeated round trips to server
- No Flicker
- Leverages you system processing power
- User actions in the browser cause JavaScript to execute
- Script communicates directly with the server
  - XmlHttpRequest or hidden IFRAME
- Server replies
  - Data: text, JSON, XML, etc.
  - HTML fragment
  - JavaScript in the page interprets this reply and uses it to update one or more page areas


New Portlet Project

Advanced Settings

Define the advanced settings of the portlet.

Single Sign-On
- [ ] Add credential vault handling
  - [● Use portlet private credential vault slot]
  - Use shared credential vault slot
  - Use administrative credential vault slot
  - Use system credential vault slot
  - Slot name: VaultSlot
  - [ ] Show password

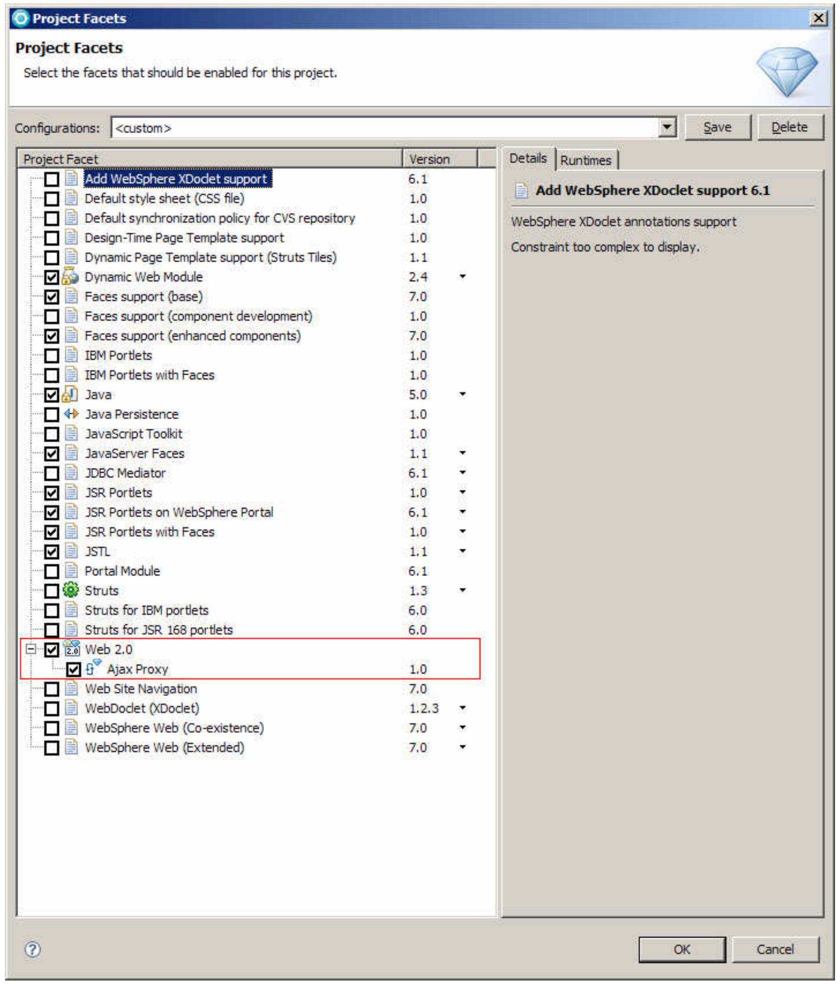
Business Process
- [ ] Enable business process integration
  - [● Add task processing]
  - Add process initiation
  - Template name: TemplateName

Dynamic Cache Key Generation
- [ ] Add cache key helper

Web2.0 Portlet Feature
- [✓] Add Client Side Model

Ajax Proxy Support

- 3 ways to add Ajax support
  - When you create a New Portlet Project
  - Through Portlet deployment Descriptor page of project
  - Through Project Facets wizard
- Easy to edit
  - Proxy Tab in Portlet Deployment Descriptor
- Easy to remove



Project Facets

Select the facets that should be enabled for this project.

Configurations: <custom>

	Project Facet	Version
☐	Add WebSphere XDoclet support	6.1
☐	Default style sheet (CSS file)	1.0
☐	Default synchronization policy for CVS repository	1.0
☐	Design-Time Page Template support	1.0
☐	Dynamic Page Template support (Struts Tiles)	1.1
☑	Dynamic Web Module	2.4
☑	Faces support (base)	7.0
☐	Faces support (component development)	1.0
☑	Faces support (enhanced components)	7.0
☐	IBM Portlets	1.0
☐	IBM Portlets with Faces	1.0
☑	Java	5.0
☐	Java Persistence	1.0
☐	JavaScript Toolkit	1.0
☑	JavaServer Faces	1.1
☐	JDBC Mediator	6.1
☑	JSR Portlets	1.0
☑	JSR Portlets on WebSphere Portal	6.1
☑	JSR Portlets with Faces	1.0
☑	JSTL	1.1
☐	Portal Module	6.1
☐	Struts	1.3
☐	Struts for IBM portlets	6.0
☐	Struts for JSR 168 portlets	6.0
☑	Web 2.0	
☑	Ajax Proxy	1.0
☐	Web Site Navigation	7.0
☐	WebDoclet (XDoclet)	1.2.3
☐	WebSphere Web (Co-existence)	7.0
☐	WebSphere Web (Extended)	7.0

Details

Add WebSphere XDoclet support 6.1

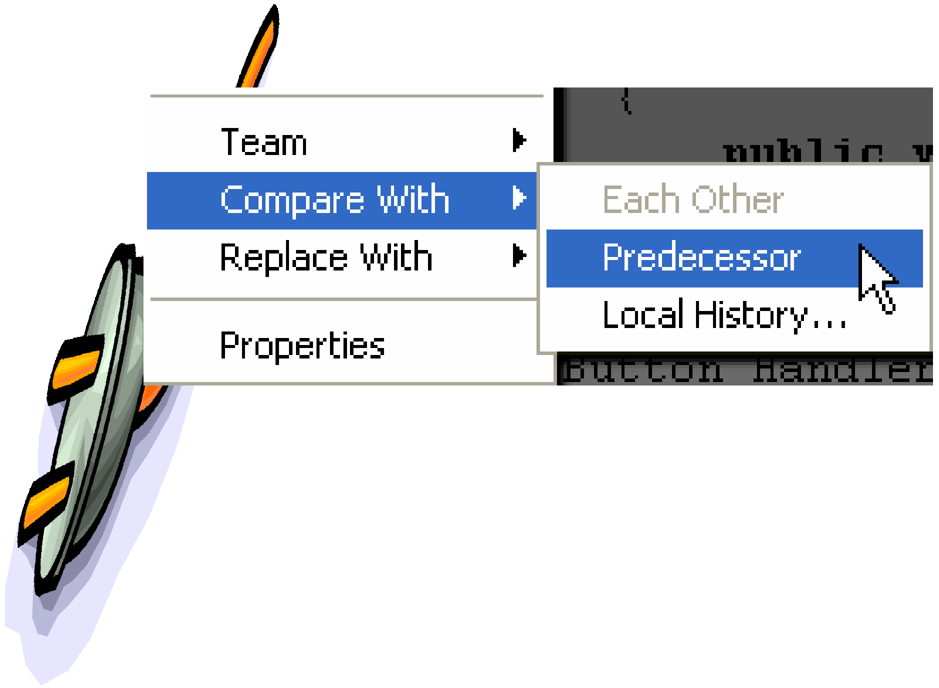
WebSphere XDoclet annotations support

Constraint too complex to display.

Extra Rational Integration content

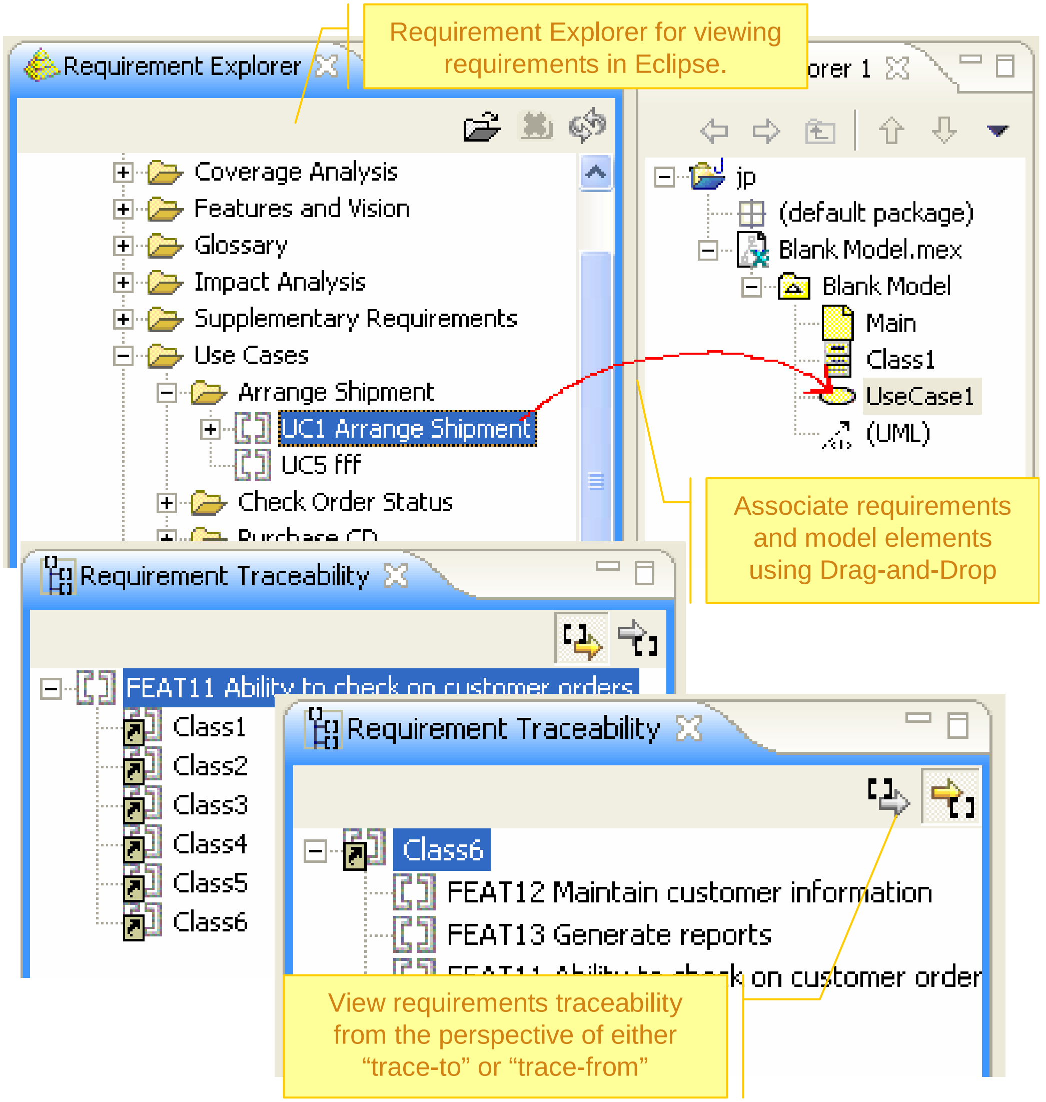
Solution: ClearCase SCM Adapter

- **Full Dynamic View support**
  - File system notification
- **Compare/merge support**
  - Integrated with Eclipse compare/merge framework
- **Disconnected Mode**
  - Manual Disconnect
- **Workspace / view management**
  - Support for workspace switching
- **Setup & Getting Started**
- **Best practices and online help improvements**



Lifecycle Integration - RequisitePro

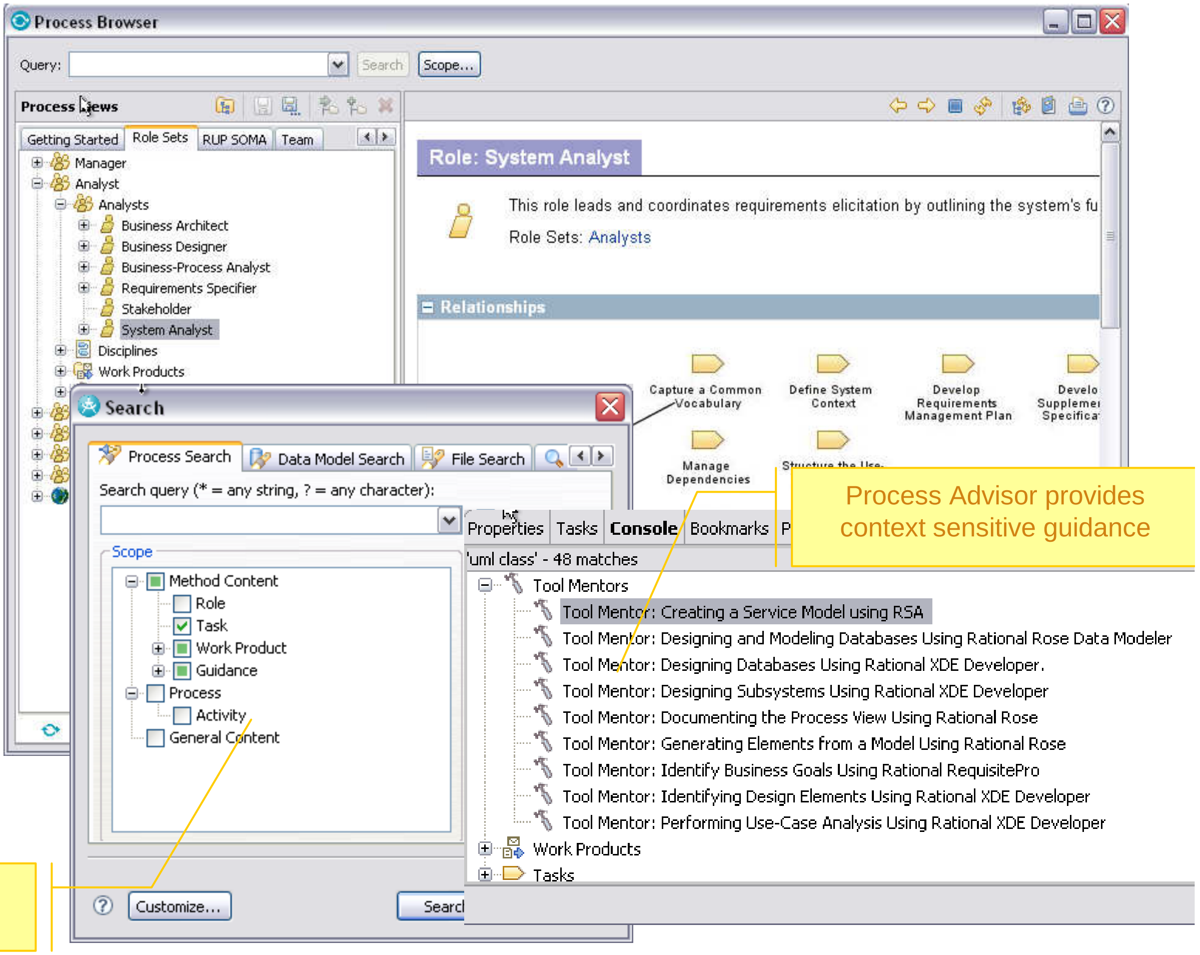
- **Traceability links established from the requirements stage through the design stage**
  - Assist users in querying RequisitePro for traceability relationships between the requirements and the Java code.
- **Requirements perspective for browsing requirements in Rational RequisitePro software and creating links to code elements**
  - Simplifies the creation of traceability links from the requirements stage through the design stage
  - Open and browse multiple RequisitePro projects
  - See requirements, packages, and views
  - Associate requirements with java classes via drag and drop
  - Create code elements from requirements
  - Customizable synchronization


Requirement Explorer for viewing requirements in Eclipse.
Requirement Explorer
Coverage Analysis
Features and Vision
Glossary
Impact Analysis
Supplementary Requirements
Use Cases
Arrange Shipment
UC1 Arrange Shipment
UC5 fff
Check Order Status
Purchase CD
orer 1
jp
(default package)
Blank Model.mex
Blank Model
Main
Class1
UseCase1
(UML)
Associate requirements and model elements using Drag-and-Drop
Requirement Traceability
FEAT11 Ability to check on customer orders
Class1
Class2
Class3
Class4
Class5
Class6
Requirement Traceability
Class6
FEAT12 Maintain customer information
FEAT13 Generate reports
FEAT11 Ability to check on customer order
View requirements traceability from the perspective of either “trace-to” or “trace-from”

Process Guidance

- **Rational Unified Process for Developers provides dynamic process guidance**
  - Process guidance and user assistance is provided dynamically as the user works with the tool.
- **Tool Mentors provide guidance for activities**
- **User customizable views with user defined content**

Improved navigation of RUP


Process Browser
Query:
Search
Scope...
Process Views
Getting Started
Role Sets
RUP SOMA
Team
Manager
Analyst
Analysts
Business Architect
Business Designer
Business-Process Analyst
Requirements Specifier
Stakeholder
System Analyst
Disciplines
Work Products
Role: System Analyst
This role leads and coordinates requirements elicitation by outlining the system's fu
Role Sets: Analysts
Relationships
Capture a Common Vocabulary
Define System Context
Develop Requirements Management Plan
Develop Supplementary Specifica
Manage Dependencies
Structure the Use-
Search
Process Search
Data Model Search
File Search
Search query (* = any string, ? = any character):
Scope
Method Content
Role
Task
Work Product
Guidance
Process
Activity
General Content
Customize...
Search
Properties
Tasks
Console
Bookmarks
'uml class' - 48 matches
Tool Mentors
Tool Mentor: Creating a Service Model using RSA
Tool Mentor: Designing and Modeling Databases Using Rational Rose Data Modeler
Tool Mentor: Designing Databases Using Rational XDE Developer.
Tool Mentor: Designing Subsystems Using Rational XDE Developer
Tool Mentor: Documenting the Process View Using Rational Rose
Tool Mentor: Generating Elements from a Model Using Rational Rose
Tool Mentor: Identify Business Goals Using Rational RequisitePro
Tool Mentor: Identifying Design Elements Using Rational XDE Developer
Tool Mentor: Performing Use-Case Analysis Using Rational XDE Developer
Work Products
Tasks

Process Advisor provides context sensitive guidance

Search is integrated with Eclipse search

Backup

Why use EJB 3.0? What's different from EJB 2.1.?

The major differences between EJB 2.x and EJB 3.0 versions:-

- Annotations are used in EJB 3.0
- Removal of home interface enabled simple lookup process in EJB 3.0
- EJB deployment descriptors are not required in EJB 3.0
- JPA entity beans don't have home and remote interfaces.
- JPA entity beans becomes local. Remote annotations are not supported for entity beans
- EJB 3.0 beans don't implement the standard interfaces like javax.ejb.SessionBean and hence no need to implement the container call back methods like ejbActivate(), etc
- Query is very flexible. Multiple levels of joins are enabled through the refined EJB-QL
- Security can be provided either through annotations or through deployment descriptors

Why use EJB 3.0? What's different from EJB 2.1.?

	EJB 2.1	EJB 3.0

Class	public class AccountBean implements javax.ejb.SessionBean
	{
	    SessionContext ctx; DataSource accountDB;
	    public void setSessionContext(SessionContext ctx) {
	        this.ctx = ctx;
	    }
	    public void ejbCreate() {
	        accountDB = (DataSource)ctx.lookup(
	"jdbc/accountDB");
	    }
	    public void ejbActivate() { }
	    public void ejbPassivate() { }
	    public void ejbRemove() { }
	    public void setAccountDeposit(int empId, double deposit) {
	        ...
	        Connection conn = accountDB.getConnection();
	        ...
	    } ...
	}
		@Stateless
		public class AccountBean implements IAccount {
		    @Resource private DataSource accountDB;
		    public void setAccountDeposit(int empId, double deposit) {
		        ...
		        Connection conn = accountDB.getConnection();
		        ...
		    } ...
		}

Side files	IAccount.class AccountBean.class IAccountHome.class IAccountLocal.class IAccountLocalHome.class	IAccount.class AccountBean.class

Why use EJB 3.0? What’s different from EJB 2.1.?

	EJB 2.1	EJB 3.0

Deployment Descriptor	`<session>`
	`  <ejb-name>AccountBean</ejb-name>`
	`  <local-home>IAccountHome</local-home>`
	`  <local>Account</local>`
	`  <ejb-class>com.example.AccountBean</ejb-class>`
	`  <session-type>Stateless</session-type>`
	`  <transaction-type>Container</transaction-type>`
	`  <resource-ref>`
	`    <res-ref-name>jdbc/accountDB</res-ref-name>`
	`    <res-ref-type>javax.sql.DataSource</res-ref-type>`
	`    <res-auth>Container</res-auth>`
	`  </resource-ref>`
	`</session>`
	...
	`<assembly-descriptor>...</assembly-descriptor>`
		Not required

Lookup	`try {`
	`  InitialContext ctx = new InitialContext();`
	`  IAccount account =`
	`      (IAccount) ctx.lookup("java:comp/env/ejb/IAccount");`
	`} catch (NamingException ne) {`
	`}`
		…
		`@EJB`
		`IAccount account;`
		…

Web 2.0 Development with IBM WebSphere sMash

Matthew Perrins
Executive IT Specialist
Software Group Lab Services
matthew_perrins@uk.ibm.com

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

IBM

Rational. software

What is WebSphere sMash?

WebSphere sMash is an Agile Web Application Platform

- Architected around Dynamic Scripting, REST, Rich Web Interfaces, AJAX, and Feeds
- Optimized for
  - **Speed**
  - **Simplicity**
  - **Agility**

Key Scenarios

- Enables developers to build web 2.0-style applications by easily pulling in, composing, and “cobbling together” pre-existing assets (PHP assets, services, feeds, code snippets) using dynamic scripting languages and simple consumption principles based on REST.
- Leverages existing SOA investments by enabling rapid development of dynamic web applications that are assembled from enterprise assets and publicly available APIs.

WHAT IS PROJECT ZERO?

Project Zero is the development and incubation community for WebSphere sMash

- Live on the Internet since June 2007

Project Zero represents

- The people that build and use WebSphere sMash
- The incubation of new technology that will deliver in future versions of WebSphere sMash
- The community of 3rd party assets that leverage the WebSphere sMash platform

All released versions are called WebSphere sMash

WEBSPHERE SMASH AND THE WEB 2.0 STRATEGY

WebSphere sMash fits perfectly into WebSphere's overall SOA Web 2.0 strategy

The WebSphere Web 2.0 Strategy

- **Unleash enterprise content so it is more easily accessible**
  - WebSphere is REST-enabling its product portfolio, including
    - MQ, Commerce, WSRR, Web 2.0 FP, WPS, WESB, Datapower,
- **Leverage this content by enabling agile web applications**
  - Now that the content is unleashed, you can agilely build web applications to leverage that content as well as content from other backend systems, supplier systems or the web.
  - WebSphere sMash fits here, along with Lotus Mashups and InfoSphere MashupHub, as an agile platform for application creation and deployment
  - WebSphere sMash can also be used here to build and deliver components such as widgets for Lotus Mashups, or feeds for InfoSphere MashupHub
- **Run, manage and host agile applications**
  - Application-centric runtimes (like WebSphere sMash) and management systems (like WebSphere XD) will help you cost effectively run and manage these growing number of these web applications.

PACKAGING

The technology is WebSphere sMash is available in a variety of packages

Package Name	Description
WebSphere sMash	Production version of the WebSphere sMash Platform. Standard IBM commercial license. Available through normal IBM channels.
WebSphere sMash Reliable Transport Extensions	Production version of the extended features in sMash related to messaging and reliable communications. Standard IBM commercial license. Available through normal IBM channels.
WebSphere sMash Developer Edition (DE)	This is the community version of the exact same code you get with WebSphere sMash. WebSphere sMash DE represents the shipped and stable version of the product for developers to use to build applications.
Project Zero	This is the community version of the latest and greatest unreleased technology that is not in a WebSphere sMash version yet. This is the bleeding edge incubation of new features.

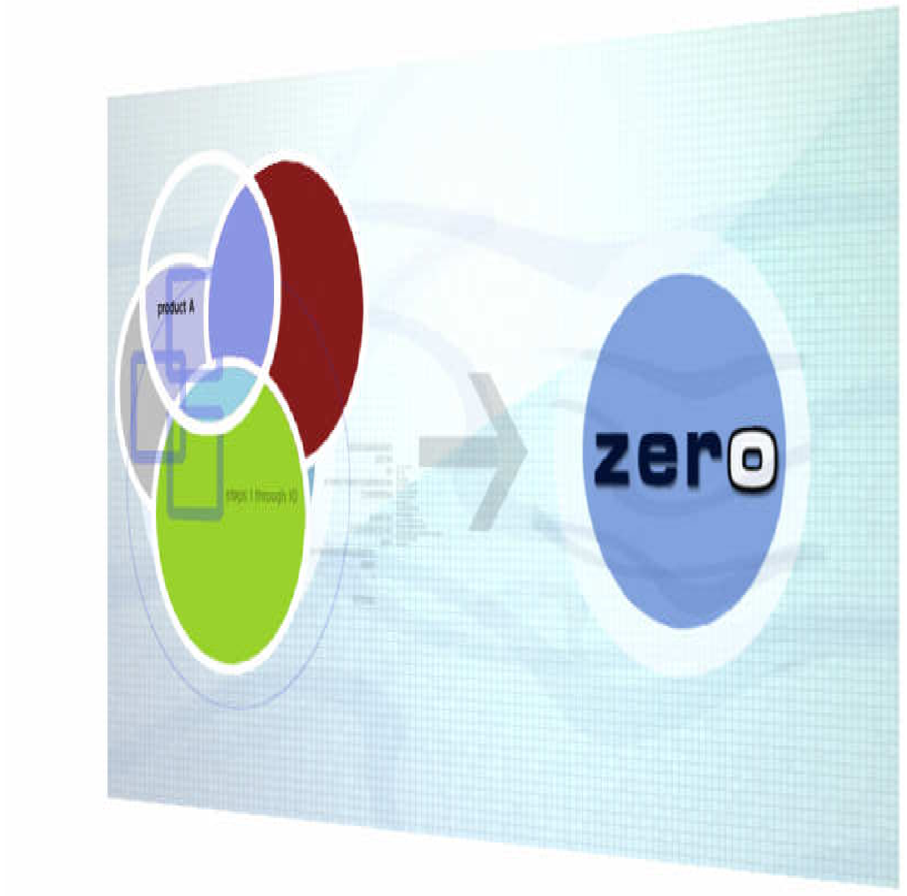
Technology Overview

THE WEBSPHERE SMASH CORE VALUES

Speed

Simplicity

Agility



CORE APPLICATION CONSTRUCTS
EASY FOR DEVELOPERS TO ACCESS, LEARN, AND USE

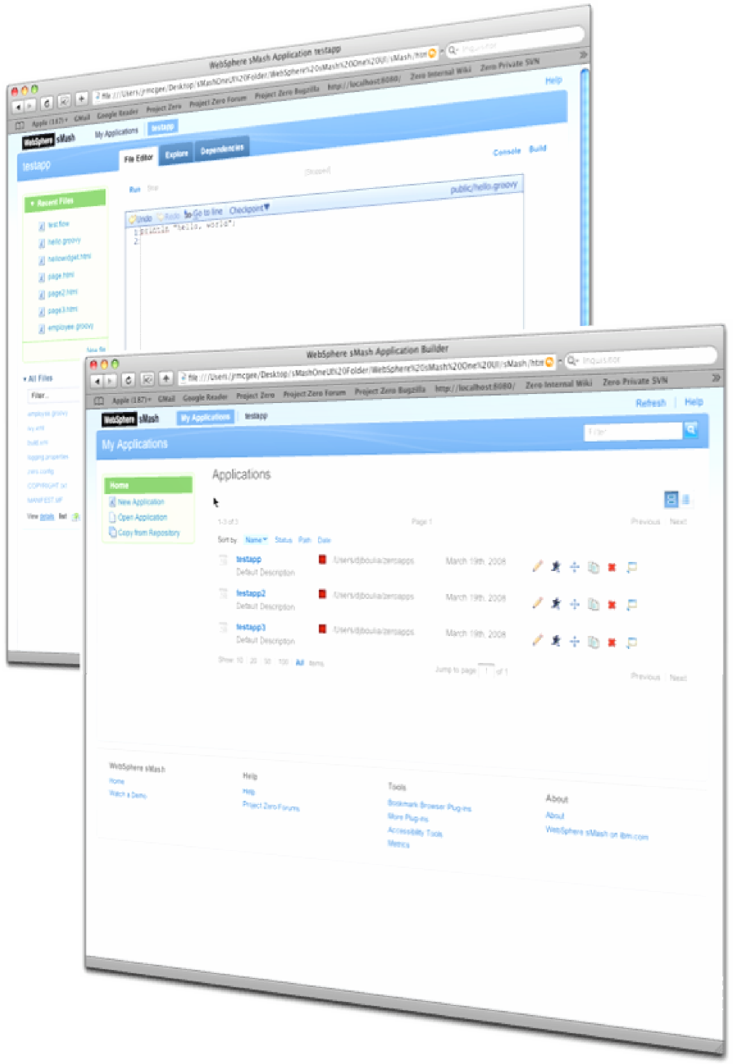
Dynamic Scripting and Templates

Effortless creation of RESTful Services and Data Feeds (RSS, ATOM)

Simple Event-based execution environment

State externalized into a shared memory space (Global Context)

Repository of pre-built Services and Libraries provides useful building blocks



DYNAMIC SCRIPTING

WebSphere sMash is a dynamic scripting platform

Application Logic is created in one of two scripting languages

- Groovy (for people that prefer Java)
- PHP (for the 3 Million existing PHP programmers)

Java is positioned as the “system” language

- Mostly used to implement system extensions and application libraries
- Entire applications can be written in Java, if desired
  - Requires more configuration

APPLICATION CENTRIC RUNTIME

WebSphere sMash is an application-centric runtime

- You create an application and run it
- You do not package an application and deploy it to a multi-application server
- Each application runs in its own process (JVM)
- Runtime is designed to be short lived
  - Update recycles after idle timeout or max number of requests

WebSphere sMash is a full stack runtime

- Everything needed to run the application is built in
  - Including the HTTP stack
- No external proxy or web server is required
- Does not deploy as a WAR file inside another JEE container
- An external proxy is used for clustering and multi-app routing

PHP SUPPORT

Gartner predict that within 5 years 60% of the 5.5 million PHP programmers will work in corporate IT. (Up from 13% of 3M today).

The PHP runtime is built on top of a standard JVM

- Supports use of many PHP Extensions
    - XAPI-C interface allows C-based extensions
    - XAPI-J interface allows Java based extensions
- Supports bridging between Java and PHP
- Currently supports a subset of PHP
    - The goal is maximum re-use of existing PHP scripts

PHP runtime provided directly by WebSphere sMash, not php.net

The goal of PHP support is about unleashing the 3 Million PHP programmers together with the vast library of existing PHP script code and extensions and bringing it to the sMash programming model

A number of popular PHP application now run on the sMash PHP runtime, including

- **phpBB**
- **SugarCRM**

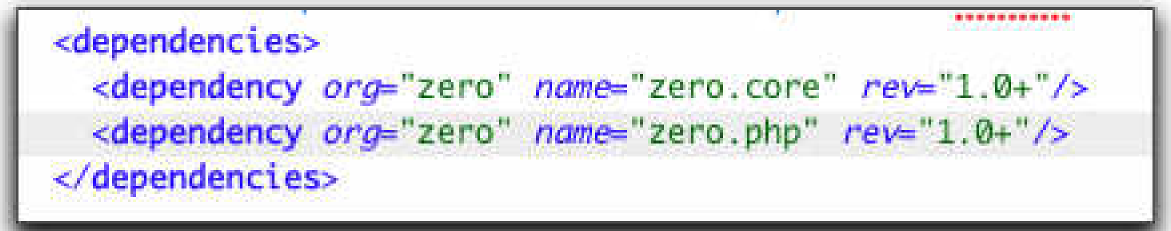
MODULAR ARCHITECTURE

WebSphere sMash applications are based on a very small core

- 5.4 MBytes (includes Groovy).
  - PHP adds additional 14.5 Mbytes
    - Contains 3 platforms currently
- Core provides all of the framework and runtime support, including HTTP transport

Additional features provided in downloadable modules

- Applications declare a dependency on desired features (using Ivy)
- A package management system manages your dependencies, including:
  - The ability to share dependencies on a machine
  - The ability to demand load missing dependencies from the network
  - The ability to manage updates to dependencies that you are using



```
<dependencies>
  <dependency org="zero" name="zero.core" rev="1.0+"/>
  <dependency org="zero" name="zero.php" rev="1.0+"/>
</dependencies>
```

SIMPLE DEPLOYMENT

Essentially the deployment is Zip and Copy

No machine specific information bound into the application

Default mode is shared dependencies

- Application dependencies are load for the deployment machines local repository and pulled off the network if needed

Standalone mode is supported as well

- All application dependencies are included in the ZIP and nothing is needed on the target machine except a JVM

Provides a packaging command to simplify the creation of the ZIP file for deployment

- zero package for shared mode
- zero package standalone for standalone mode

RUNTIME CHARACTERISTICS

Instant On

- Application Available for Service in less than 1 sec
  - 0.672 seconds on a MacBook Pro
- Application JVM starts in about 1 second
  - 1.3 seconds on a MacBook Pro

Clean

- Graceful recovery, isolation, tolerates “bad” code
- Short lived processes
  - Runs for fixed number of requests or idle timeout then restarts
  - No state lost on restart

Cheap

- Cost effective to run in small and large quantities
- Idle Application Footprint ~380 Kbytes
- Running Application JVM ~28 Mbytes TODAY

Supported on “stock” JVM

- IBM, Sun, Mac, etc - Any JSE 5 or 6 JVM

BUILT-IN DEVELOPMENT TOOLING

WEBSPERE SMASH LETS DEVELOPERS BUILD APPLICATION DIRECTLY ON THE WEB

Browser-Based Development IDE

Built as a sMash application

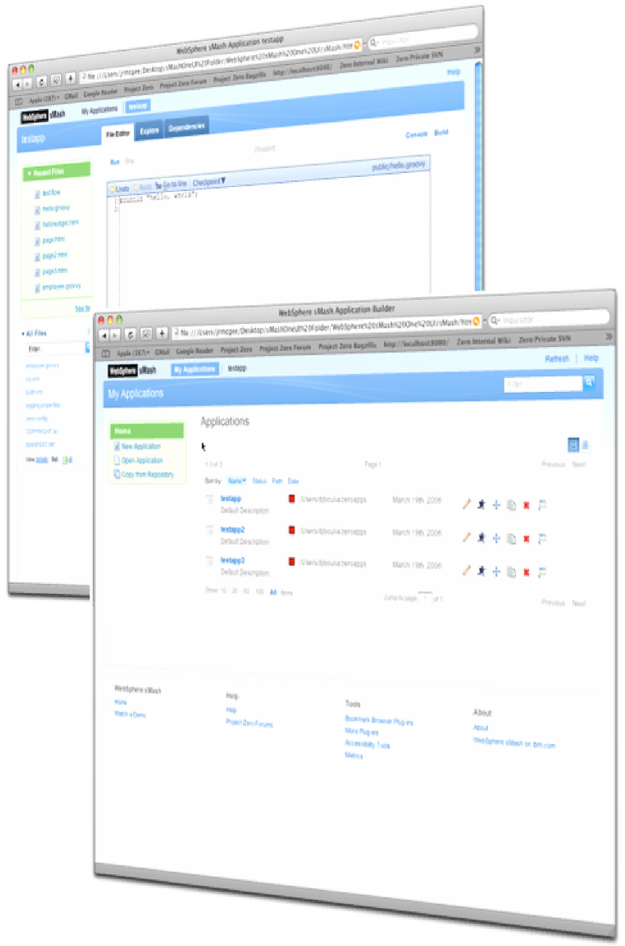
Provides full development lifecycle for Zero applications

- Create, Edit, Test

Provides Visual Editors for Activities and Web Page construction

- Including a DOJO-enabled page editor

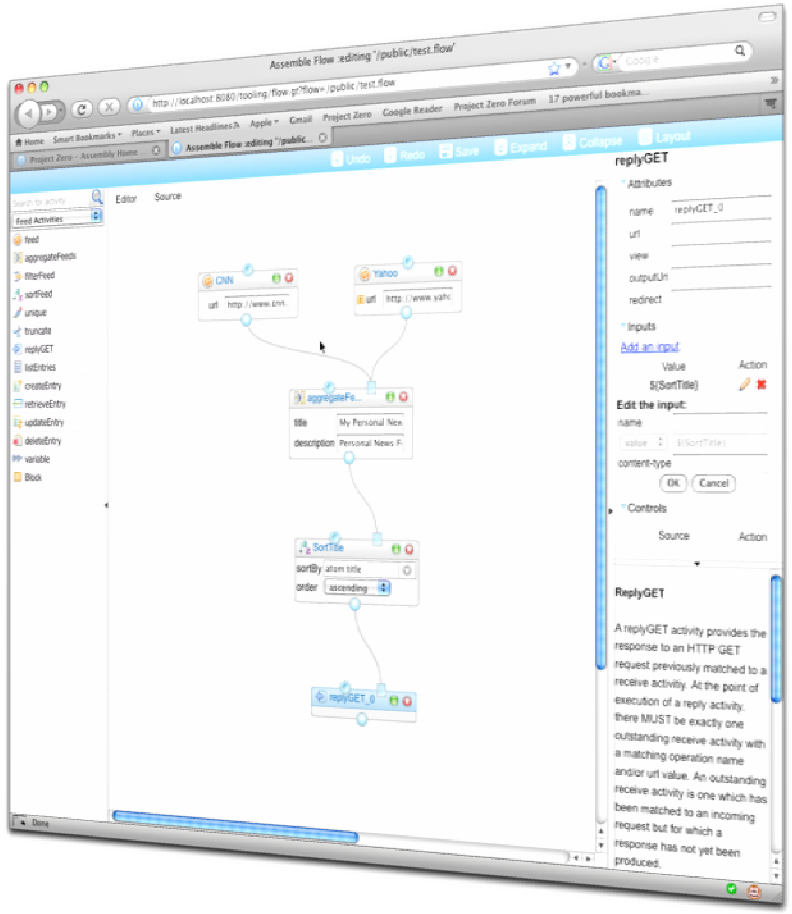
Basic Eclipse-based tooling also available if required



WEBSPHERE SMASH ACTIVITIES
LETS DEVELOPERS VISUALLY "MASH-UP" SERVICES AND FEEDS

Assemble-style Development

- Compose applications by "wiring" together REST services
- Visually or programmatically combine existing feeds and services that enrich, sort, and filter data in a pipeline
- Configure templates to alter pipeline routes, log events along the pipeline
- Numerous built-in activities, including
  - Get Feed, Call Service, Aggregate, Sort, Transform, Filter, Send Mail, XSLT, Conditionals, Loops

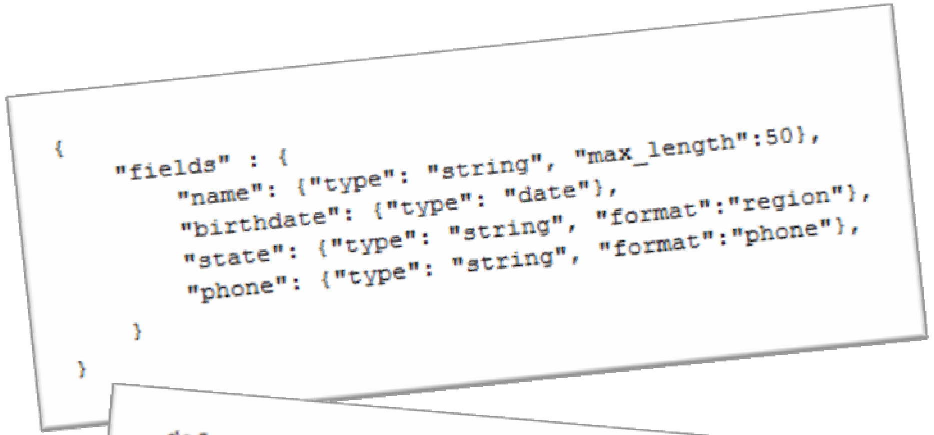


ReplyGET

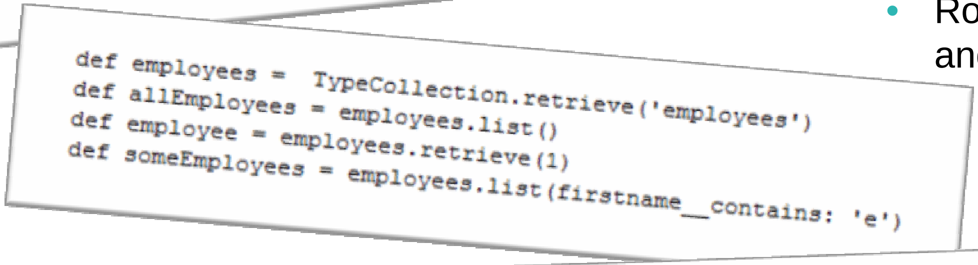
A replyGET activity provides the response to an HTTP GET request previously matched to a receive activity. At the point of execution of a reply activity, there MUST be exactly one outstanding receive activity with a matching operation name and/or url value. An outstanding receive activity is one which has been matched to an incoming request but for which a response has not yet been produced.

RAPIDLY EXPOSE DATA RESTFULLY

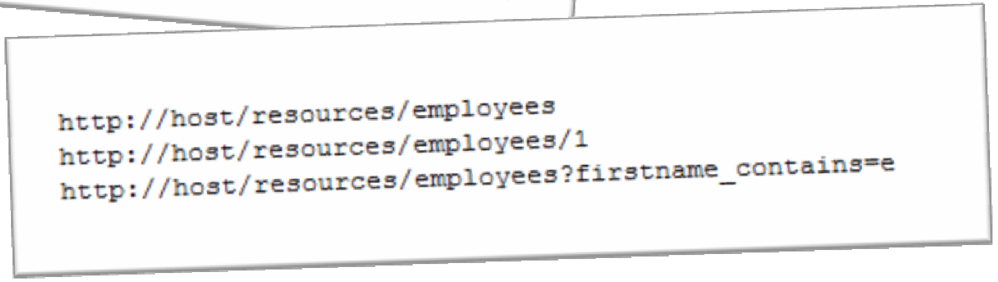
ZERO RESOURCE MODEL ENABLES DEVELOPERS WITH A SIMPLE PROGRAMMATIC AND HTTP DATA API



```
{
    "fields" : {
        "name": {"type": "string", "max_length":50},
        "birthdate": {"type": "date"},
        "state": {"type": "string", "format":"region"},
        "phone": {"type": "string", "format":"phone"},
    }
}
```



```
def employees =  TypeCollection.retrieve('employees')
def allEmployees = employees.list()
def employee = employees.retrieve(1)
def someEmployees = employees.list(firstname__contains: 'e')
```



```
http://host/resources/employees
http://host/resources/employees/1
http://host/resources/employees?firstname_contains=e
```

Model application data

- Constrained set of APIs encourage a RESTful application architecture
- Data model that maps well into Atom feeds and JSON formats
- Robust framework for persistence, validation, and serialization

AVAILABLE MODULES

There are approximately 65 modules available currently

Modules provide function in many categories

- Data Formats (JSON, ATOM, RSS, XML)
- Data Access
- Resource Modeling
- Security / Content Filtering
- Activity Flows
- Services
  - Amazon ECS, Flickr, Weather, etc
- Utilities (such as HTML parsing)
- Management Tools
- Development Tooling
- Reliable Transport Engine for Messaging Interactions

COMMUNITY DRIVEN COMMERCIAL DEVELOPMENT

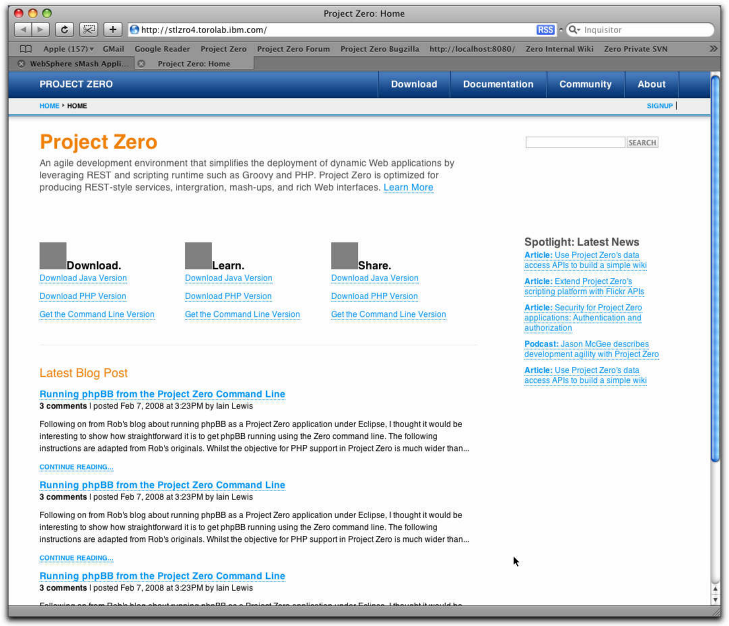
Evolve the core platform based on developer feedback

Commercial development using a transparent development process

Enabled via an external web site providing:

- *A focal point for all sMash development activities*
  - Expose the IBM development process to the external developer community
  - All design decisions are discussed and communicated via external forums
  - Registered users can post comments and feedback to the forums
- *Frictionless download of latest code and documentation*
  - Registration not required for binary downloads
  - Latest builds immediately available to developers
  - Source code can be viewed by registered users

http://www.projectzero.org



Project Zero

An agile development environment that simplifies the deployment of dynamic Web applications by leveraging REST and scripting runtime such as Groovy and PHP. Project Zero is optimized for producing REST-style services, intergration, mash-ups, and rich Web interfaces. Learn More

Download.	Learn.	Share.
Download Java Version	Download Java Version	Download Java Version
Download PHP Version	Download PHP Version	Download PHP Version
Get the Command Line Version	Get the Command Line Version	Get the Command Line Version

Spotlight: Latest News

Article: Use Project Zero's data access APIs to build a simple wiki

Article: Extend Project Zero's scripting platform with Flickr APIs

Article: Security for Project Zero applications: Authentication and authorization

Podcast: Jason McGee describes development agility with Project Zero

Article: Use Project Zero's data access APIs to build a simple wiki

Latest Blog Post

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

Following on from Rob's blog about running phpBB as a Project Zero application under Eclipse, I thought it would be interesting to show how straightforward it is to get phpBB running using the Zero command line. The following instructions are adapted from Rob's originals. Whilst the objective for PHP support in Project Zero is much wider than...

CONTINUE READING...

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

Following on from Rob's blog about running phpBB as a Project Zero application under Eclipse, I thought it would be interesting to show how straightforward it is to get phpBB running using the Zero command line. The following instructions are adapted from Rob's originals. Whilst the objective for PHP support in Project Zero is much wider than...

CONTINUE READING...

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

PROJECTZERO.ORG
LOWERING THE BARRIERS TO COMMUNITY ACCESS

Anonymous Visitors can…

Browse the site
View Wiki content
Read Forums
Search the Bug Database
Read Blogs
Download Binary Drivers*

Focused on easy access

- Internet web site
- Free access to the platform

Registered Users can…

Post to the Forum
Submit Bug Reports
Submit Feature Requests
Comment on Blog Posts
Access Source Code*

Focused on feedback

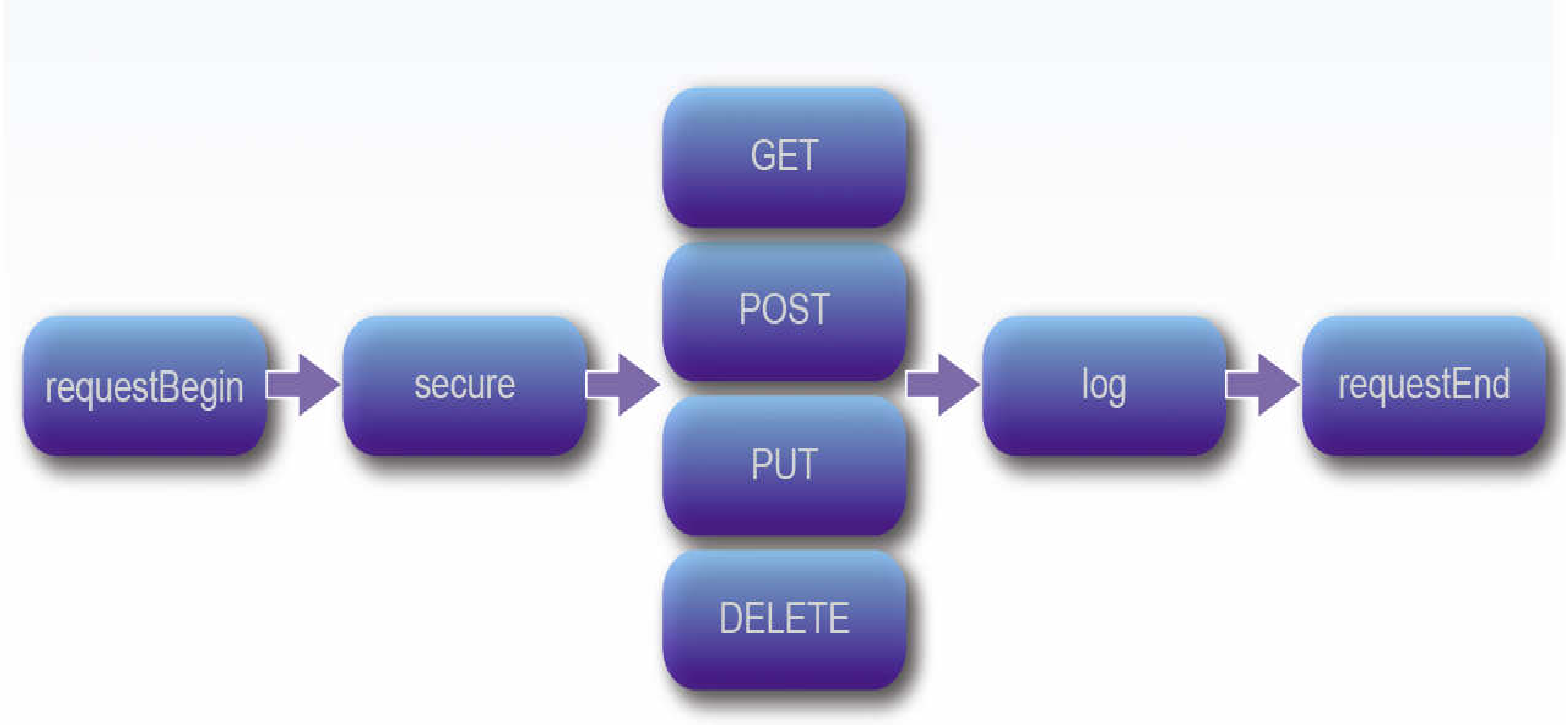
- Simple, free registration process

* Requires acceptance of an IBM license agreement

Core Programming Model

Events

- All behavior in the system is modeled as a set of event
  - Applications are built by handling these events and providing desired behavior
  - Similar to AJAX model or classic UI programming



Event Handlers

All handlers are stateless

Can be implemented in Groovy, PHP, and Java

Groovy

```
println "Hello World"
```

```
def onGET()
{
   println "Hello World"
}
```

PHP

```
function onGET()
{
   echo "Hello World";
}
```

Java

```
public void onGET()
{
   PrintWriter writer = (PrintWriter)zget("/request/writer");
   writer.println("Hello World");
}
```

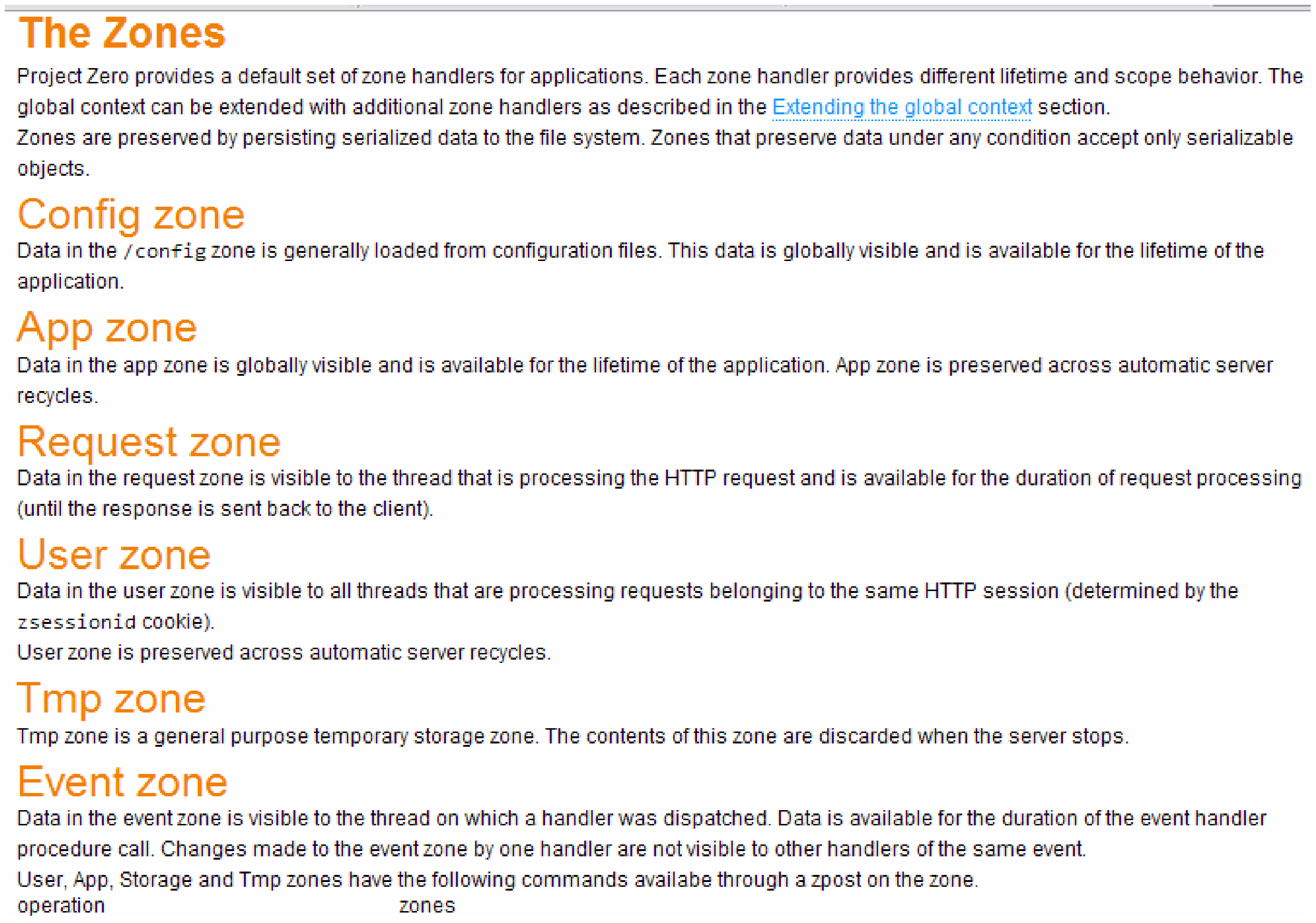
```
/config/handlers += {
   "events" : "GET",
   "handler" : "HelloWorld.class",
   "conditions" : ["/request/path matches /hello"]
}
```

Global Context – State Management

- **The Global Context (GC) provides access to and management of all application state**
  - Conceptually a map of data
- **Externalizes all state from the application logic**
  - Enables the restart ability of the JVM without data loss
  - Enables clustering and scaling to be added transparently
- Simplifies and unifies access to application state and data structures and simplifies state passing within the application
- Contains information provided by both the runtime (such as request parameters) and by the application

Global Context Zone

Divided into zones representing different data lifecycles



The Zones

Project Zero provides a default set of zone handlers for applications. Each zone handler provides different lifetime and scope behavior. The global context can be extended with additional zone handlers as described in the Extending the global context section.

Zones are preserved by persisting serialized data to the file system. Zones that preserve data under any condition accept only serializable objects.

Config zone

Data in the `/config` zone is generally loaded from configuration files. This data is globally visible and is available for the lifetime of the application.

App zone

Data in the app zone is globally visible and is available for the lifetime of the application. App zone is preserved across automatic server recycles.

Request zone

Data in the request zone is visible to the thread that is processing the HTTP request and is available for the duration of request processing (until the response is sent back to the client).

User zone

Data in the user zone is visible to all threads that are processing requests belonging to the same HTTP session (determined by the `zsessionid` cookie).

User zone is preserved across automatic server recycles.

Tmp zone

Tmp zone is a general purpose temporary storage zone. The contents of this zone are discarded when the server stops.

Event zone

Data in the event zone is visible to the thread on which a handler was dispatched. Data is available for the duration of the event handler procedure call. Changes made to the event zone by one handler are not visible to other handlers of the same event.

User, App, Storage and Tmp zones have the following commands availabe through a zpost on the zone.

operation	zones

Accessing the Global Context

Data is organized by a URI structure

- First part of URI is always the Zone name
  - /app, /user, /request, /config, /event, /client

Access is modeled after REST

- GET, PUT, POST, DELETE

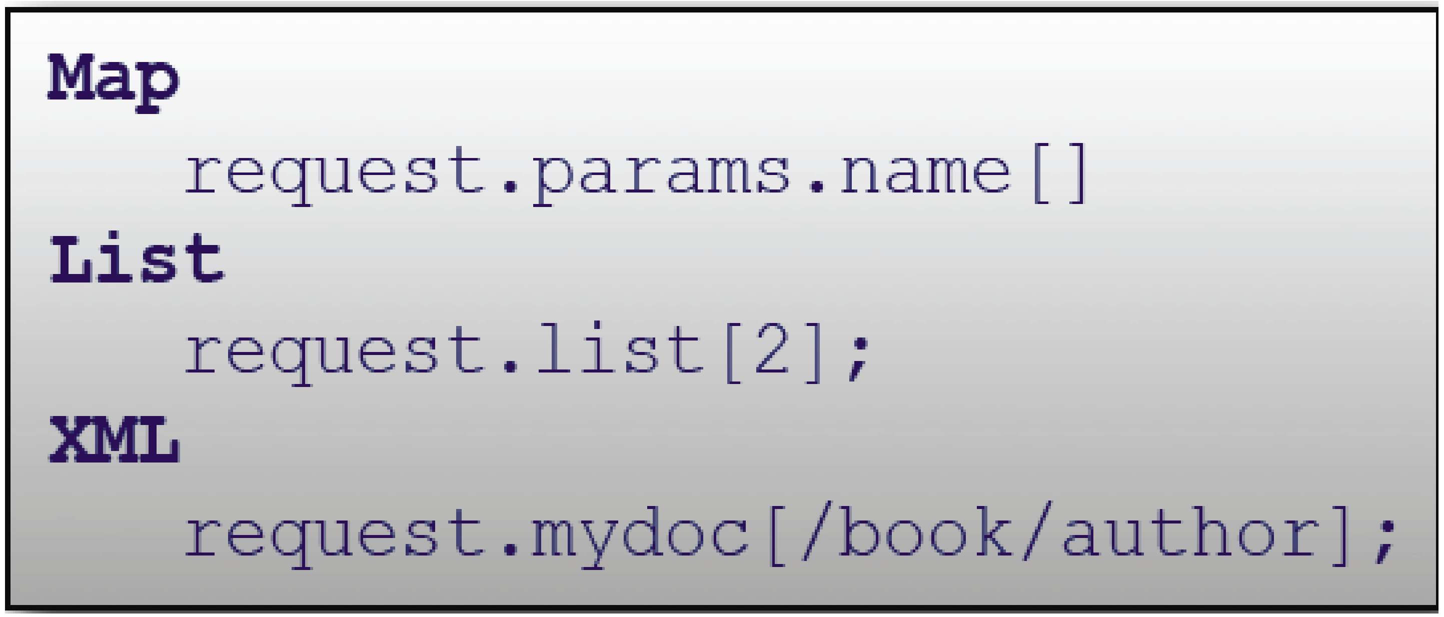
```
Java
    String path = GlobalContext.zget("/request/path");
    GlobalContext.zput("/user/counter", i);
```

```
PHP
    $path = zget("/request/path");
    zput("/user/counter", $i);
```

```
Groovy (zget/zput work too)
    def path = request.path[];
    user.counter = i;
```

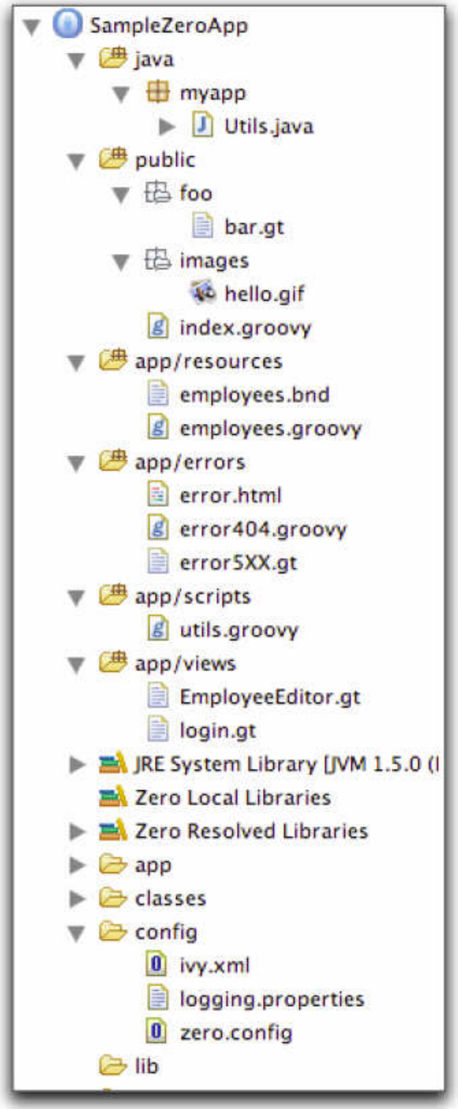
Value Pathing

- The GC provides simplified access to certain data structures
  - Called ***Value Pathing***
- Understands
  - Maps, List, Objects, XML, JSON
- □Allows read and write access to internals of the structure through the GC address

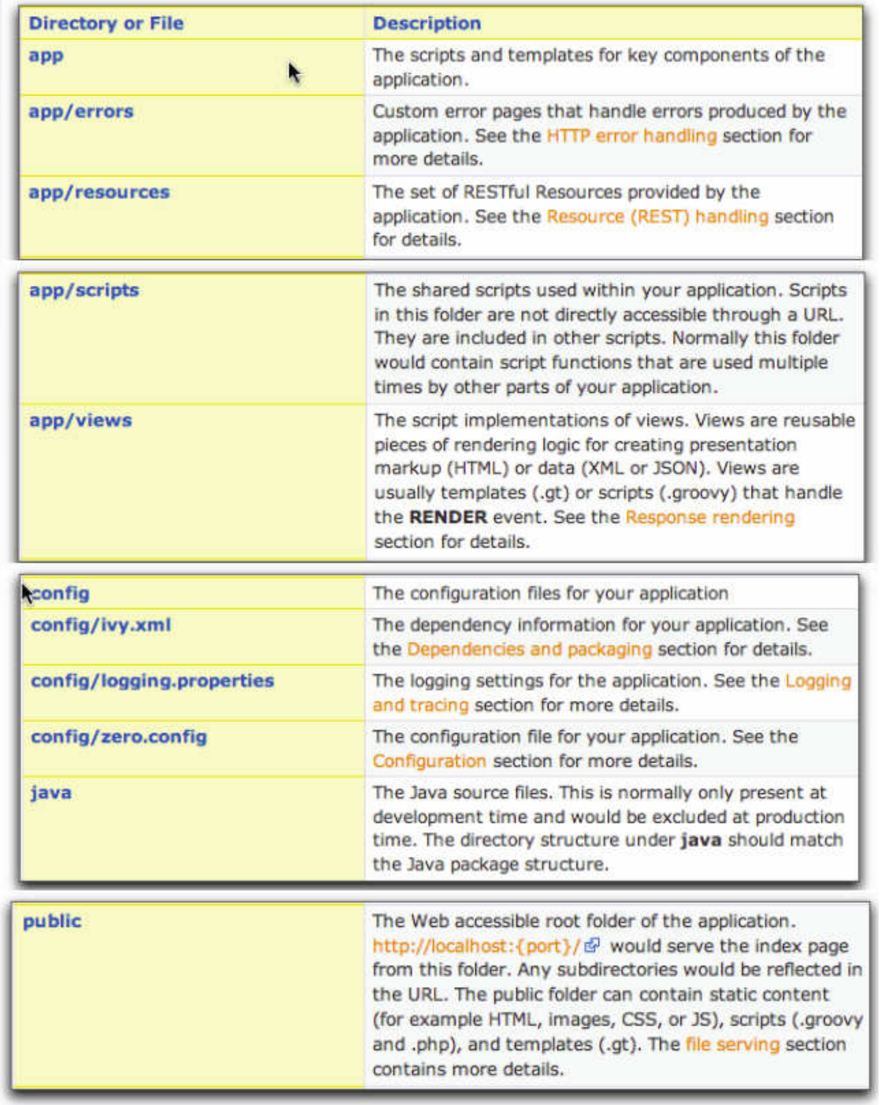


```
Map
    request.params.name[]
List
    request.list[2];
XML
    request.mydoc[/book/author];
```

Application Directory Layout



- SampleZeroApp
  - java
    - myapp
      - Utils.java
  - public
    - foo
      - bar.gt
    - images
      - hello.gif
    - index.groovy
  - app/resources
    - employees.bnd
    - employees.groovy
  - app/errors
    - error.html
    - error404.groovy
    - error5XX.gt
  - app/scripts
    - utils.groovy
  - app/views
    - EmployeeEditor.gt
    - login.gt
  - JRE System Library [JVM 1.5.0 (I
  - Zero Local Libraries
  - Zero Resolved Libraries
  - app
  - classes
  - config
    - ivy.xml
    - logging.properties
    - zero.config
  - lib

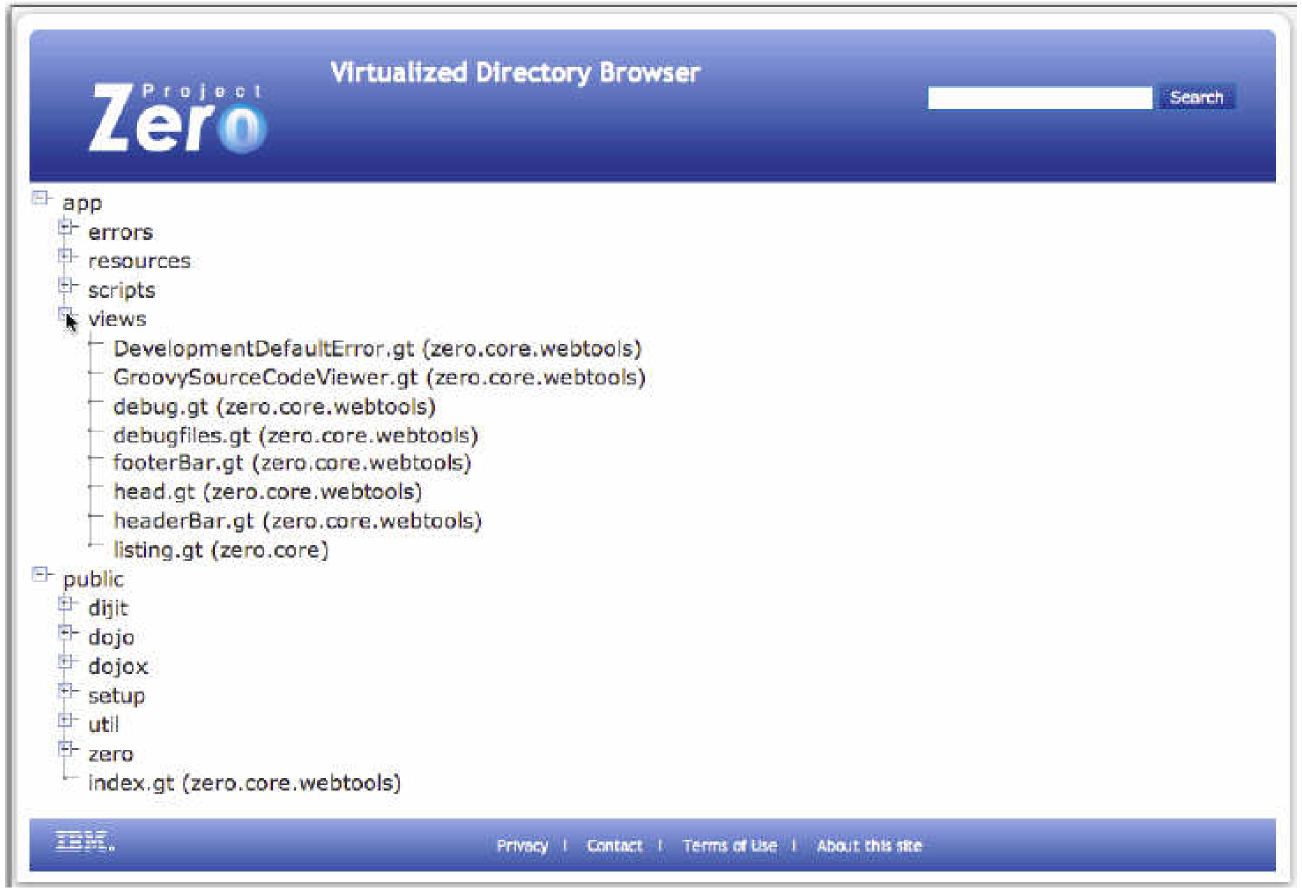


Directory or File	Description
app	The scripts and templates for key components of the application.
app/errors	Custom error pages that handle errors produced by the application. See the HTTP error handling section for more details.
app/resources	The set of RESTful Resources provided by the application. See the Resource (REST) handling section for details.
app/scripts	The shared scripts used within your application. Scripts in this folder are not directly accessible through a URL. They are included in other scripts. Normally this folder would contain script functions that are used multiple times by other parts of your application.
app/views	The script implementations of views. Views are reusable pieces of rendering logic for creating presentation markup (HTML) or data (XML or JSON). Views are usually templates (.gt) or scripts (.groovy) that handle the **RENDER** event. See the Response rendering section for details.
config	The configuration files for your application
config/ivy.xml	The dependency information for your application. See the Dependencies and packaging section for details.
config/logging.properties	The logging settings for the application. See the Logging and tracing section for more details.
config/zero.config	The configuration file for your application. See the Configuration section for more details.
java	The Java source files. This is normally only present at development time and would be excluded at production time. The directory structure under **java** should match the Java package structure.
public	The Web accessible root folder of the application. http://localhost:{port}/ would serve the index page from this folder. Any subdirectories would be reflected in the URL. The public folder can contain static content (for example HTML, images, CSS, or JS), scripts (.groovy and .php), and templates (.gt). The file serving section contains more details.

Virtualized Directory

Project Zero provides seamless integration of directories across an application and its dependencies, while maintaining each as separate entities.

All artifacts are searched within both the application and its declared dependencies



```
app
  errors
  resources
  scripts
  views
    DevelopmentDefaultError.gt (zero.core.webtools)
    GroovySourceCodeViewer.gt (zero.core.webtools)
    debug.gt (zero.core.webtools)
    debugfiles.gt (zero.core.webtools)
    footerBar.gt (zero.core.webtools)
    head.gt (zero.core.webtools)
    headerBar.gt (zero.core.webtools)
    listing.gt (zero.core)
public
  dijit
  dojo
  dojox
  setup
  util
  zero
  index.gt (zero.core.webtools)
```

Configuration

Zero configuration file: zero.config

The config/zero.config file is processed at the start of a Zero application.

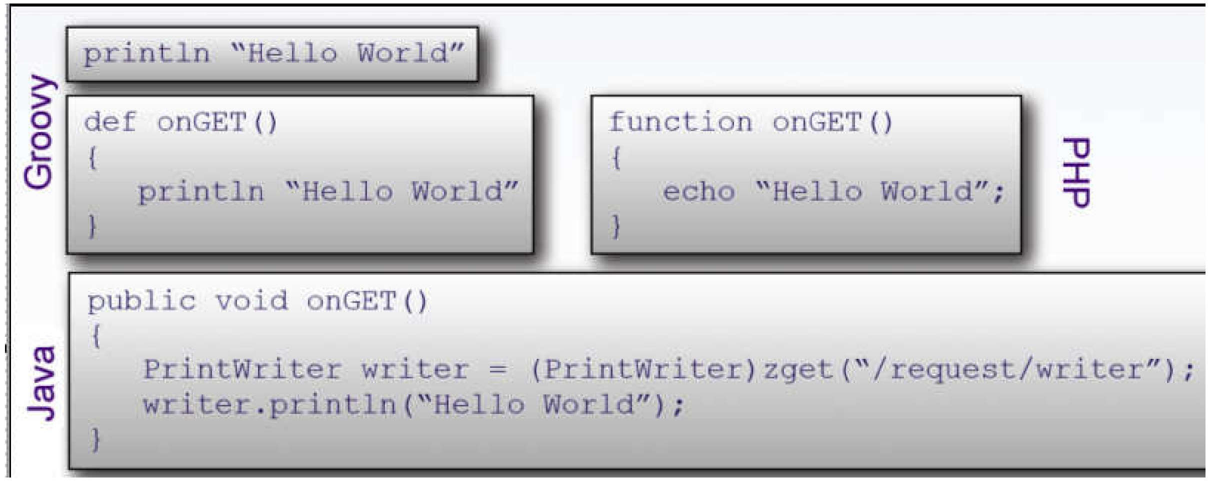
The content of a config/zero.config file is organized into "stanzas" of related key/value pairs. Stanzas are associated with directives, such as "store to the Global Context" and "include another configuration file."

```
# Value set
/config/http/port = 8080

# List set
/config/resources/defaultExtensions = [".groovy"]

# List append
/config/bindings/.groovy +=
["zero.core.groovysupport.bindings.DefaultBindings"]

# Map set
/config/test/map = { "a" : "b", "c" : "d" }

# Map append
/config/test/mapappend += { "a" : "b", "c" : "d" }
/config/test/mapappend += { "x" : "y", "w" : "z" }

# Event handler
/config/handlers += {
  "events" : "GET",
  "handler" : "custom.Handler.class"
}

# Value reference (insert value read at config-load time)
/config/property/myPrefix = "/foo/bar"
/config/test/value = "${/config/property/myPrefix}/bat"

# Variable set/value reference
myPrefix = "/foo/bar"
/config/test/value = "${myPrefix}/bat"

# Include
@include "${/config/dependencies/zero.core}/config/security/
form.config"
{ "formLoginPage" : "/login" }
```

Rendering

Direct

Indirect

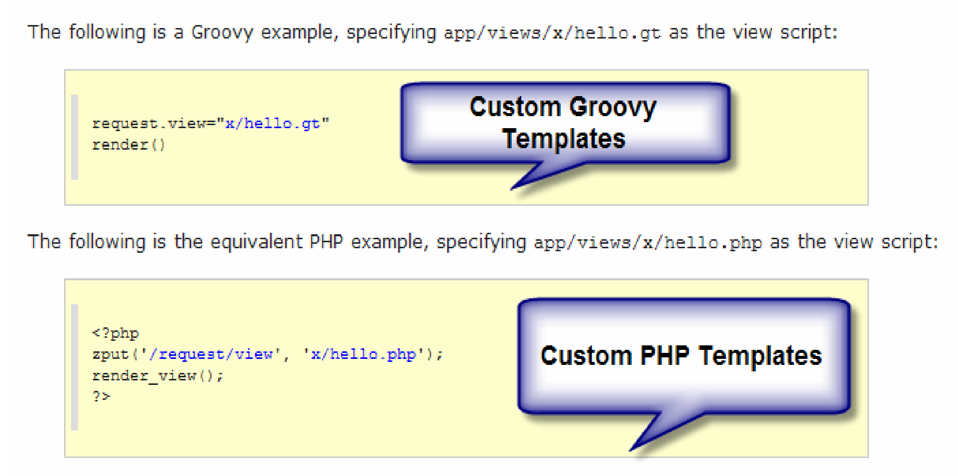


```
println "Hello World"
```

```
def onGET()
{
   println "Hello World"
}
```

```
function onGET()
{
   echo "Hello World";
}
```

```
public void onGET()
{
   PrintWriter writer = (PrintWriter)zget("/request/writer");
   writer.println("Hello World");
}
```

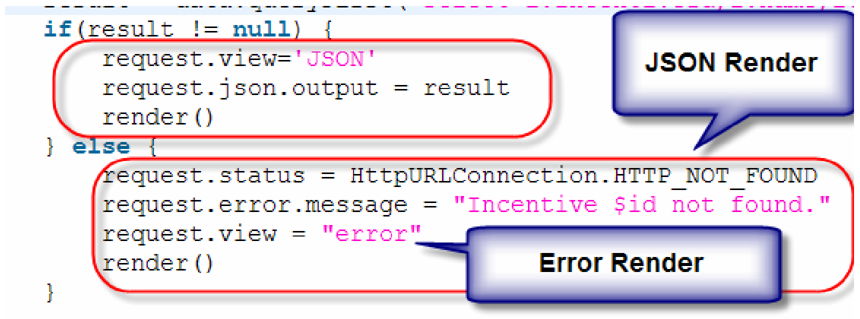


The following is a Groovy example, specifying `app/views/x/hello.gt` as the view script:

```
request.view="x/hello.gt"
render()
```

The following is the equivalent PHP example, specifying `app/views/x/hello.php` as the view script:

```
<?php
zput('/request/view', 'x/hello.php');
render_view();
?>
```



```
if(result != null) {
    request.view='JSON'
    request.json.output = result
    render()
} else {
    request.status = HttpURLConnection.HTTP_NOT_FOUND
    request.error.message = "Incentive $id not found."
    request.view = "error"
    render()
}
```

REST

What is REST?

- Representational State Transfer
- Architectural model on which the World Wide Web is based
- Principles of REST
  - Resource-centric approach
  - All relevant resources are addressable via URIs
  - Uniform access via HTTP – GET, POST, PUT, DELETE
  - Content-type negotiation allows retrieving alternative representations from same URI
- REST-style services
  - are easy to access from code running in web browsers, any other client or servers
  - can serve multiple representations of the same resource

Accessibility for Developers

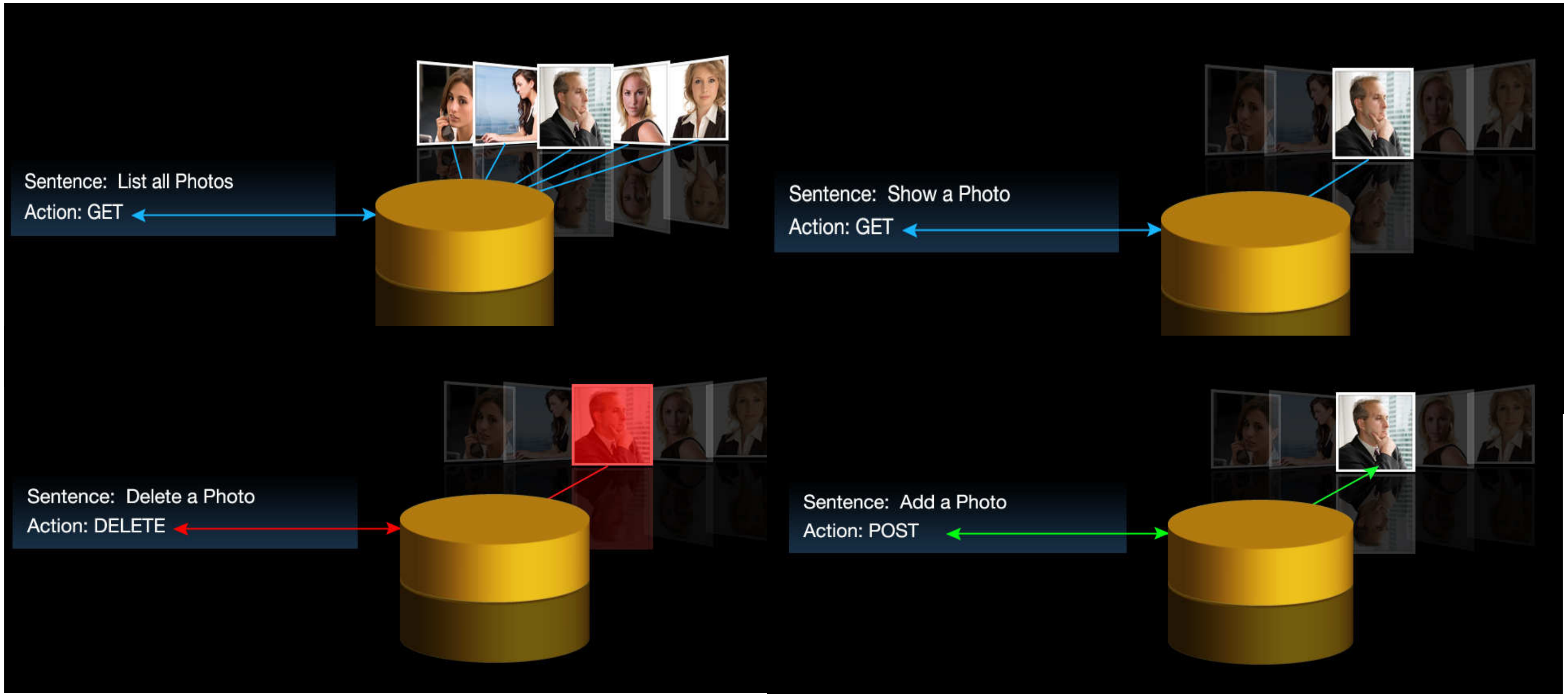
Simply exposing services from the enterprise as URLs and Feeds

A RESTful Web service is formed like a sentence:

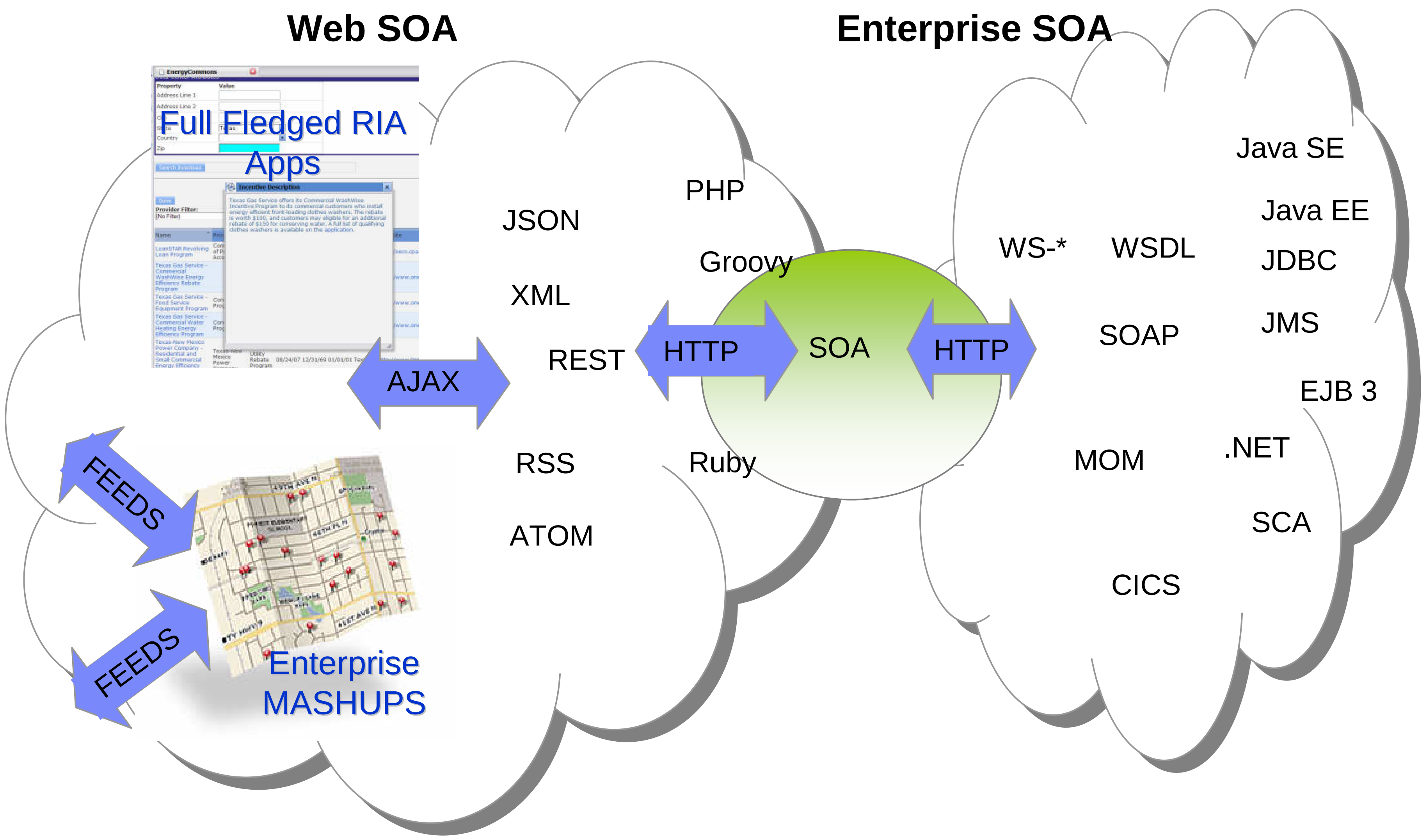
Verb = HTTP Action (GET, POST, PUT, DELETE)

Noun = the URI of the Service (the document)

Adjective = MIME type of the resulting document



Bridging Web SOA and Enterprise SOA



WebSphere sMash and REST

Resources on the Web

What are the URIs?

Which methods are supported at each URI?

What formats?

Resource	URI	Method	Representation	Description
Employee list	/resources/employee	GET	JSON (list)	List
		POST	JSON (employee)	Create
Employee	/resources/employee/{id}	GET	JSON (employee)	Retrieve
		PUT	JSON (employee)	Update
		DELETE		Delete

Representations

Employee

```
{
  "first_name“ : "Alice",
  "last_name“  : "Rogers",
  "location"   : "Seattle"
}
```

List of employees

```
[{
  "first_name“ : "Alice",
  "last_name“  : "Rogers",
  "location"   : "Seattle"
 },
 {
  "first_name“ : "Bill”,
  "last_name"  : "Stevens",
  "location"   : "Seattle“
}]
```

Resource Handlers in Zero

Basic event handlers for /resources/*

URI pattern	**Method**	**Event**	**Description**
/resources/collection	GET	list	List of all members
	POST	create	Create member
/resources/collection/{id}	GET	retrieve	Retrieve one member
	PUT	update	Replace member
	DELETE	delete	Delete member

Resource Handlers Example

app/resources/employee.groovy

```
def onList() {
  try {
    // Get configured DataManager for data access
    def data = zero.data.groovy.Manager.create('employee_db')

    // Retrieve employee records via Data Zero
    def result = data.queryArray('SELECT * FROM employees')
    request.view = 'JSON'
    request.json.output = result
    render()

  } catch (Exception e){
    if (e.getCause() instanceof java.sql.SQLException) {
      request.status = HttpURLConnection.HTTP_INTERNAL_ERROR
      request.view = 'error'
      request.error.message = 'The db may not have been initialized.'
      render()
    }
  }
} def onCreate() { ...
```

Resource Handlers Example

app/resources/employee.groovy (continued)

```
def onCreate() {
  // Convert entity to JSON object
  def emp = zero.json.Json.decode(request.input[])

  // Get configured DataManager for data access
  def data = zero.data.groovy.Manager.create('employee_db')

  // Insert employee record via Data Zero APIs
  data.update("""
    INSERT INTO employees
      (username, firstname, lastname, location, phonenumber)
    VALUES ($emp.username, $emp.firstname, $emp.lastname,
            $emp.location, $emp.phonenumber)
""")

  // Set a Location header with URI to the new record
  locationUri = getRequestedUri(false) + '/' + emp['username']
  request.headers.out.Location = locationUri
  request.status = HttpURLConnection.HTTP_NO_CONTENT
}
```

Renderers

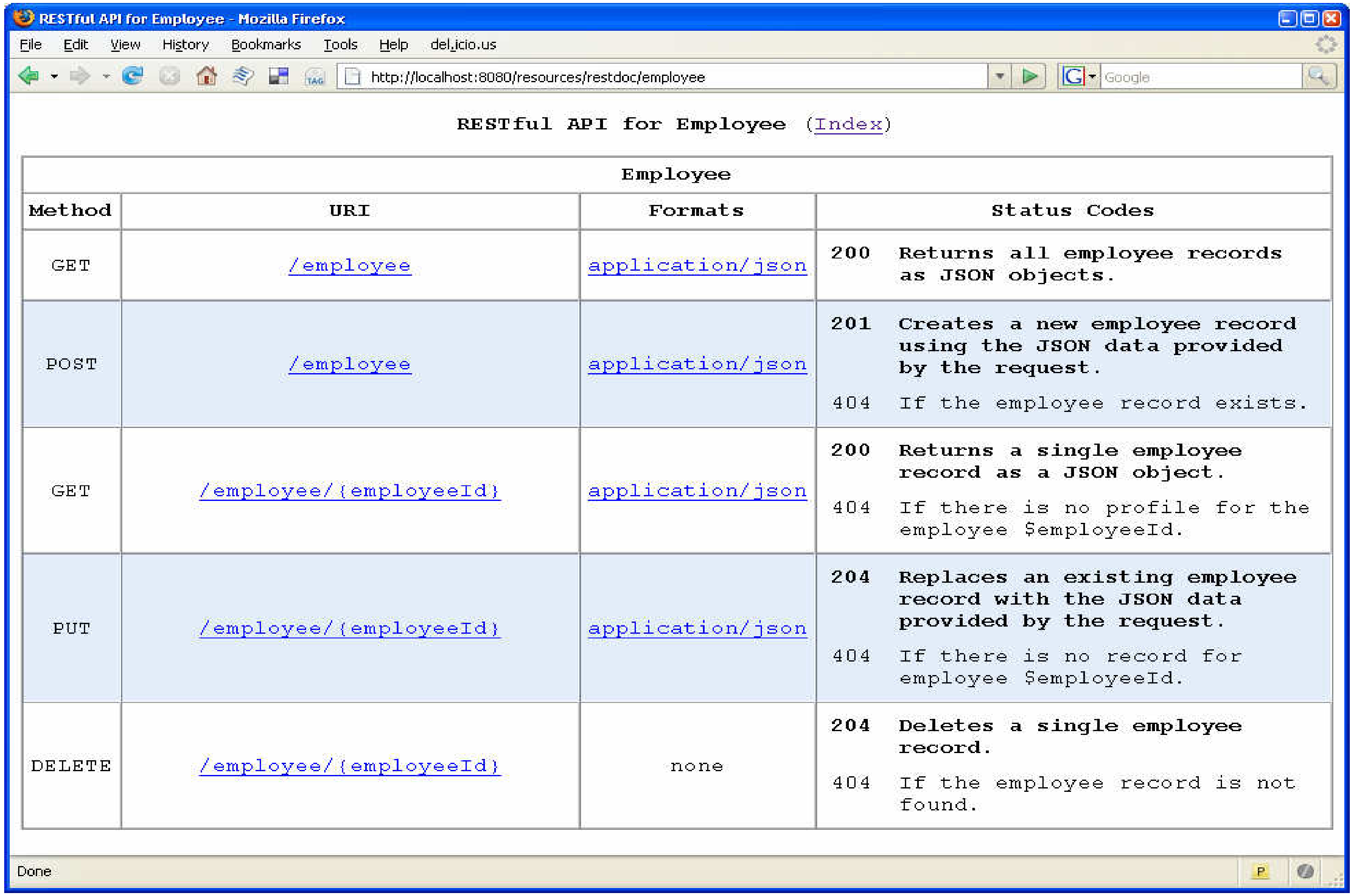
```
request.view = ‘JSON’

request.json.output = object

render()
```

JSON, XML

ATOM

Map	=> Atom entry
List<Map>	=> Atom feed

RESTdoc


RESTful API for Employee - Mozilla Firefox
File Edit View History Bookmarks Tools Help del.icio.us
http://localhost:8080/resources/restdoc/employee
Google
Done

RESTful API for Employee (Index)

Employee				
Method	URI	Formats	Status Codes	
GET	/employee	application/json	200	Returns all employee records as JSON objects.
POST	/employee	application/json	201	Creates a new employee record using the JSON data provided by the request.
			404	If the employee record exists.
GET	/employee/{employeeId}	application/json	200	Returns a single employee record as a JSON object.
			404	If there is no profile for the employee $employeeId.
PUT	/employee/{employeeId}	application/json	204	Replaces an existing employee record with the JSON data provided by the request.
			404	If there is no record for employee $employeeId.
DELETE	/employee/{employeeId}	none	204	Deletes a single employee record.
			404	If the employee record is not found.

An alternative: Zero Resource Model (ZRM)

Model application data

- Constrained set of APIs encourages a **RESTful** application architecture
- Data model that maps well into Atom feeds and JSON formats
- Robust framework for persistence, validation, and serialization

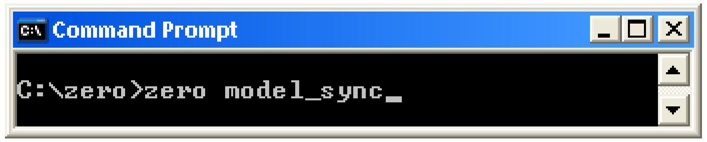
ZRM (continued)

app/models/employee.json

```
{
    "fields" : {
        "first_name": {"type":"string"},
        "last_name": {"type":"string"},
        "location": {"type":"string"}
    }
}
```

app/resources/employee.groovy

```
ZRM.delegate();
```



Command Prompt

```
C:\zero>zero model_sync_
```

app/models/fixtures/initial_data.json

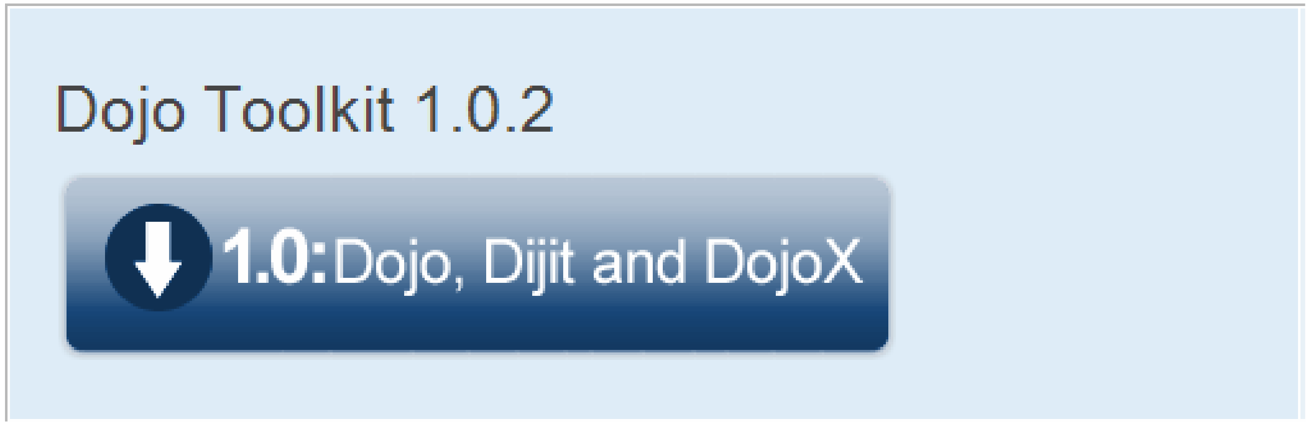
```
[
    {
        "type": "employee",
        "fields": {
                "first_name" :   "Alice",
                "last_name“  :   "Rogers",
                "location"   :   "Seattle"
        }
    },
    {
        "type": "employee",
        "fields": {
                "first_name" :   "Bill",
                "last_name"  :   "Stevens",
                "location"   :   "Seattle"
        }
    },
    {
        "type": "person",
        "fields": {
                "first_name" :   "Cathy",
                "last_name"  :   "Tomlin",
                "location"   :   "Boston"
        }
    }
]
```

WebSphere sMash and RIA

Dojo Broswer Toolkit

Dojo is an Open Source DHTML toolkit written in JavaScript. It builds on several contributed code bases.

- Provides Rich Set of Widgets
- Web UI Framework
- Rich Event handling System
- General Purpose HTML Libraries
- Several other utilities
  - Math, XML to JS parsing, etc…



Dojo Architecture

Base

The kernel of the toolkit wrapped into a **25k** js file (dojo.js). Base bootstraps the toolkit, includes AJAX utilities, class based inheritance, packaging system and more

Core

Provides addition facilities on top of the base for accessing data stores, effects such as wipes/slides, internationalization (i18n) and back-button handling among other things. Separate package keeps base small

Dijit

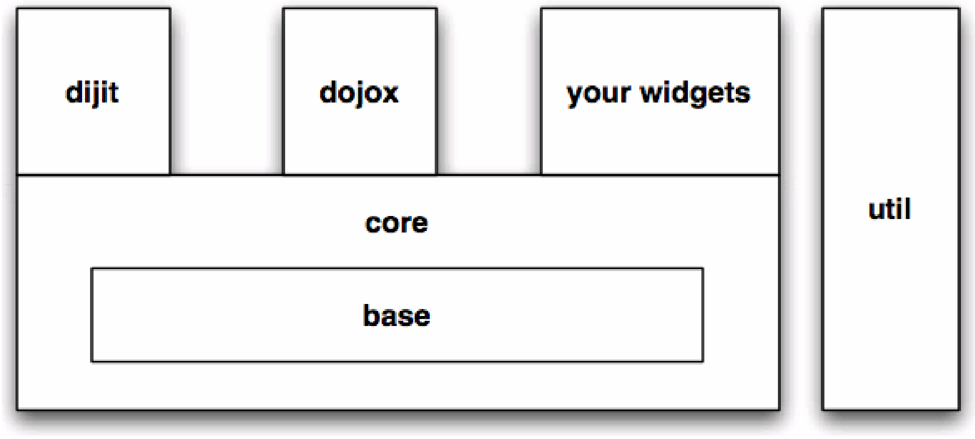
Shorthand for "Dojo widget". Could refer to a single Dojo widget (a dijit) or to the entire component of the toolkit containing all of Dojo's widgets (Dijit)

DojoX

“Dojo Extra" and contains features that stand a chance of one day migrating into Core, Dijit or even a new module. A great proving ground for new features while maintaining standards of core and base.

Util

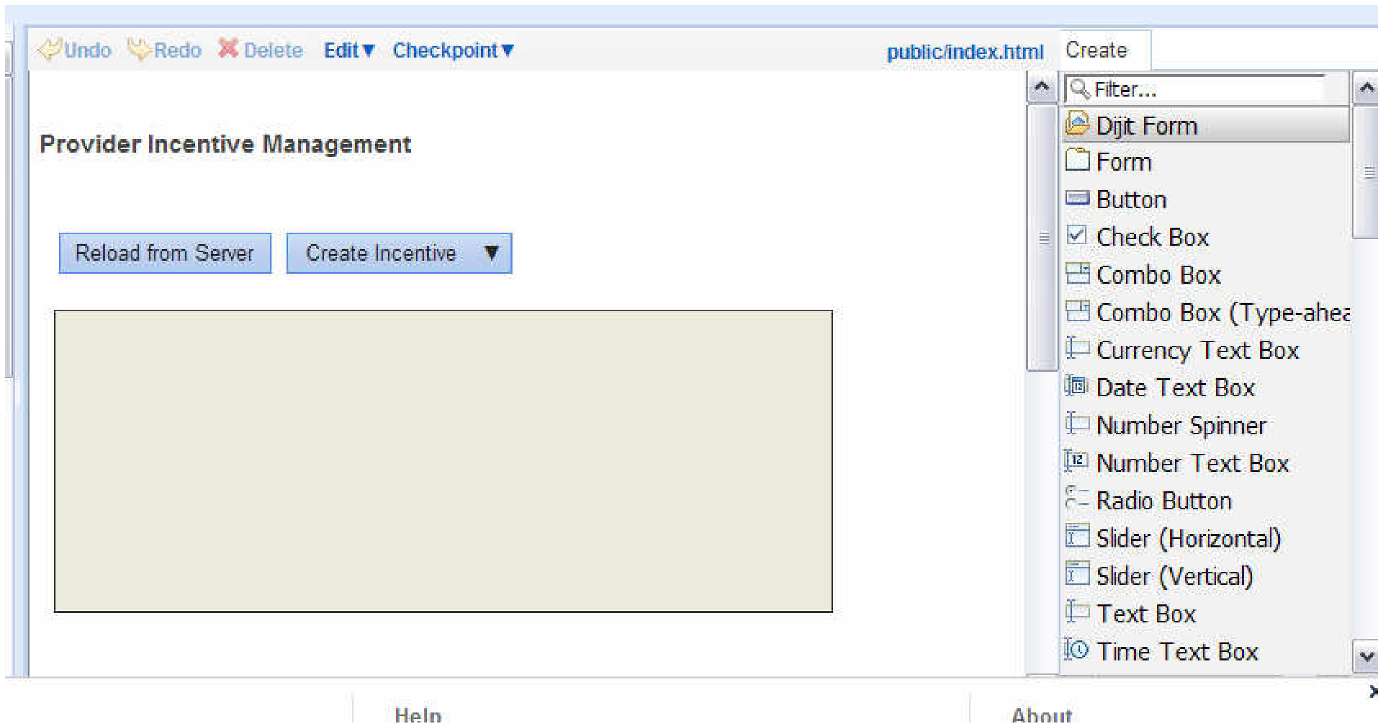
A collection of Dojo utilities (more later)



Web Based IDE Editor for Dojo

Dojo connections with Services through Wires.

Drag and Drop with Dojo Dijits.



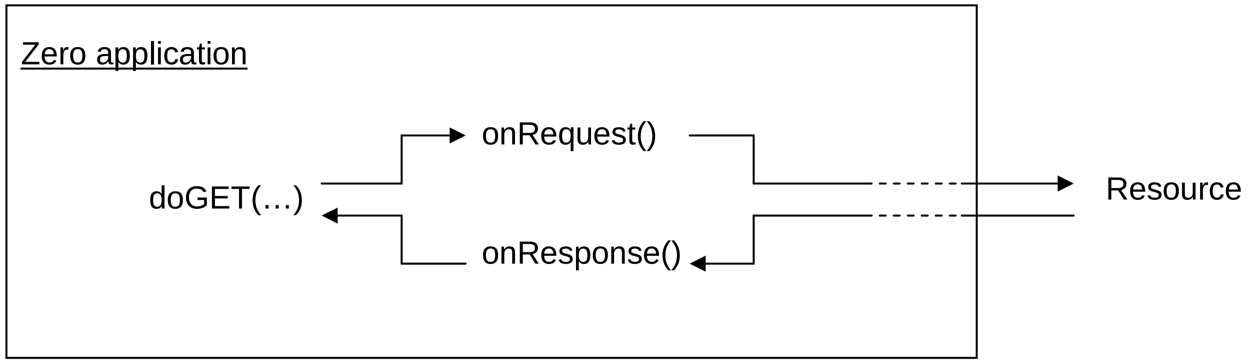
Connections

RESTful API to resources via a variety of protocols

- HTTP/S
- SMTP
- File
- JMS

In-process mediations

- SOAP
- Custom



WebSphere sMash and Flows

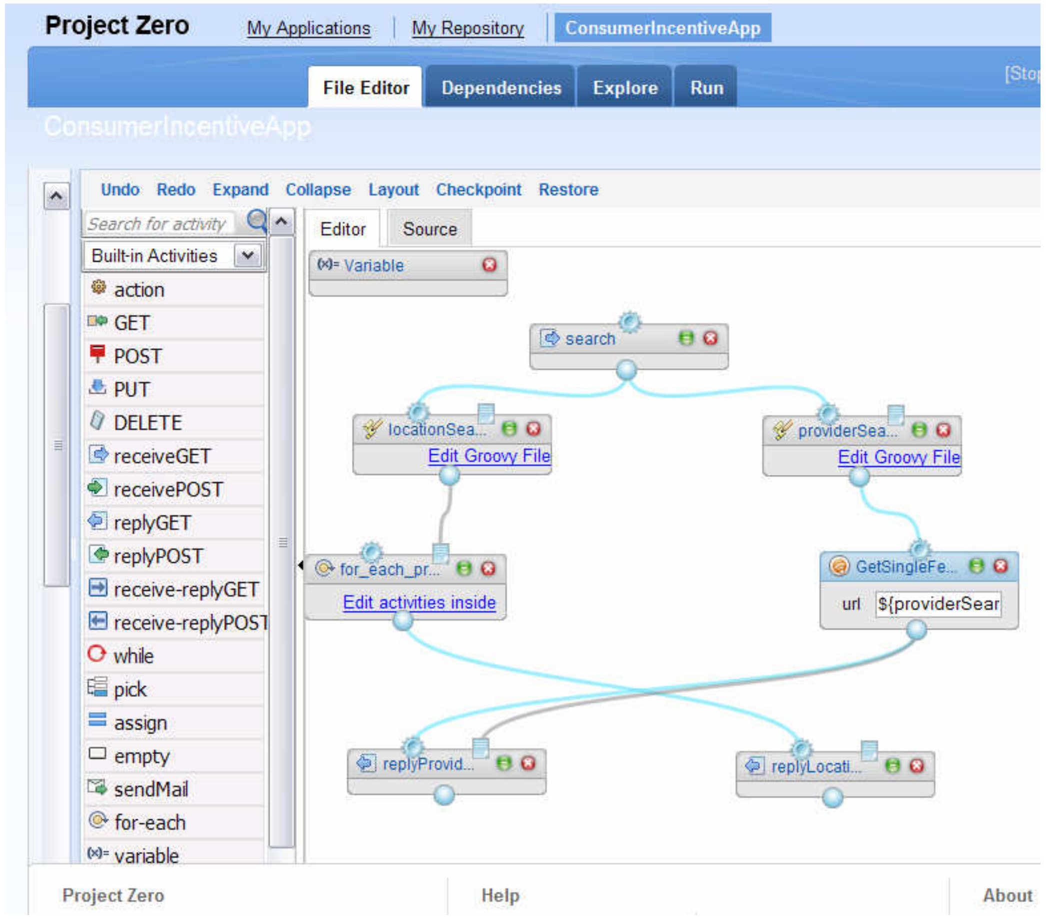
Assemble

Composition of applications by "wiring" REST services using the SPLICE flows. Incorporating both activity and data flows.

A solution may be rapidly assembled by combining existing feeds and services that enrich, sort, and filter data in a pipeline. Either visually or programmatically.

Configure templates to alter pipeline routes, log events along the pipeline, as well as transform data

Adapters to enhance integration with existing systems.



Questions & Answers

AM09

Web 2.0 Development with IBM WebSphere sMash

Matthew Perrins
Executive IT Specialist
Software Group Lab Services
matthew_perrins@uk.ibm.com

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

IBM

Rational. software

What is WebSphere sMash?

WebSphere sMash is an Agile Web Application Platform

- Architected around Dynamic Scripting, REST, Rich Web Interfaces, AJAX, and Feeds
- Optimized for
  - **Speed**
  - **Simplicity**
  - **Agility**

Key Scenarios

- Enables developers to build web 2.0-style applications by easily pulling in, composing, and “cobbling together” pre-existing assets (PHP assets, services, feeds, code snippets) using dynamic scripting languages and simple consumption principles based on REST.
- Leverages existing SOA investments by enabling rapid development of dynamic web applications that are assembled from enterprise assets and publicly available APIs.

WHAT IS PROJECT ZERO?

Project Zero is the development and incubation community for WebSphere sMash

- Live on the Internet since June 2007

Project Zero represents

- The people that build and use WebSphere sMash
- The incubation of new technology that will deliver in future versions of WebSphere sMash
- The community of 3rd party assets that leverage the WebSphere sMash platform

All released versions are called WebSphere sMash

WEBSPHERE SMASH AND THE WEB 2.0 STRATEGY

WebSphere sMash fits perfectly into WebSphere's overall SOA Web 2.0 strategy

The WebSphere Web 2.0 Strategy

- **Unleash enterprise content so it is more easily accessible**
  - WebSphere is REST-enabling its product portfolio, including
    - MQ, Commerce, WSRR, Web 2.0 FP, WPS, WESB, Datapower,
- **Leverage this content by enabling agile web applications**
  - Now that the content is unleashed, you can agilely build web applications to leverage that content as well as content from other backend systems, supplier systems or the web.
  - WebSphere sMash fits here, along with Lotus Mashups and InfoSphere MashupHub, as an agile platform for application creation and deployment
  - WebSphere sMash can also be used here to build and deliver components such as widgets for Lotus Mashups, or feeds for InfoSphere MashupHub
- **Run, manage and host agile applications**
  - Application-centric runtimes (like WebSphere sMash) and management systems (like WebSphere XD) will help you cost effectively run and manage these growing number of these web applications.

PACKAGING

The technology is WebSphere sMash is available in a variety of packages

Package Name	Description
WebSphere sMash	Production version of the WebSphere sMash Platform. Standard IBM commercial license. Available through normal IBM channels.
WebSphere sMash Reliable Transport Extensions	Production version of the extended features in sMash related to messaging and reliable communications. Standard IBM commercial license. Available through normal IBM channels.
WebSphere sMash Developer Edition (DE)	This is the community version of the exact same code you get with WebSphere sMash. WebSphere sMash DE represents the shipped and stable version of the product for developers to use to build applications.
Project Zero	This is the community version of the latest and greatest unreleased technology that is not in a WebSphere sMash version yet. This is the bleeding edge incubation of new features.

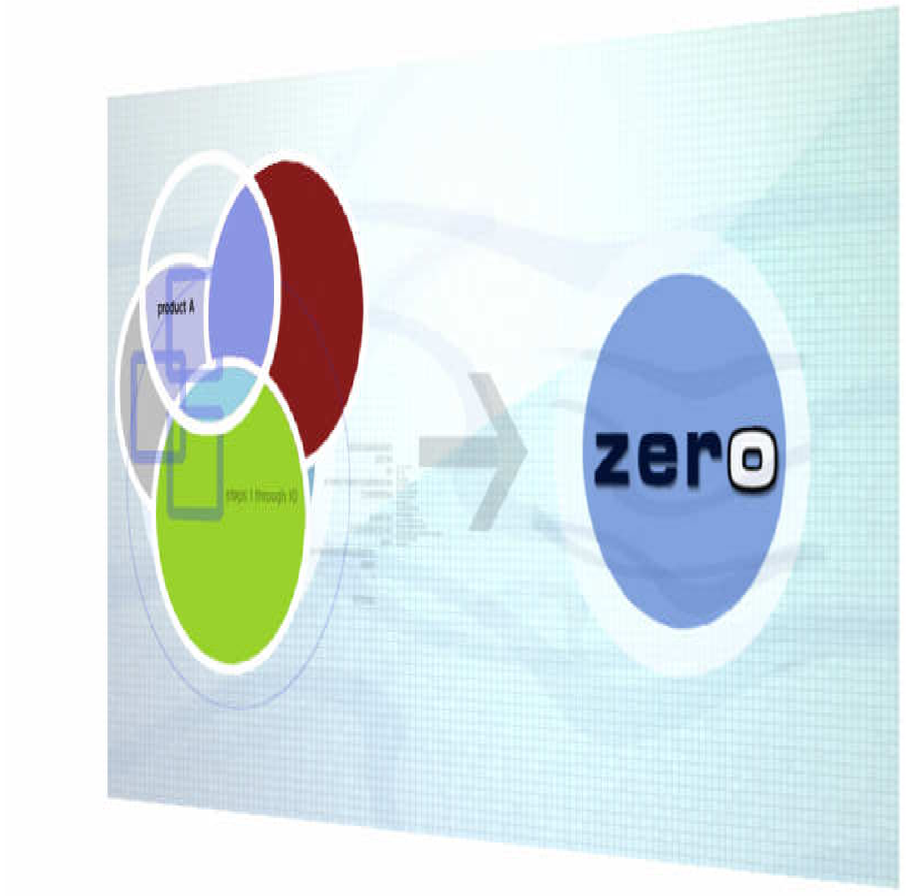
Technology Overview

THE WEBSPHERE SMASH CORE VALUES

Speed

Simplicity

Agility



CORE APPLICATION CONSTRUCTS

EASY FOR DEVELOPERS TO ACCESS, LEARN, AND USE

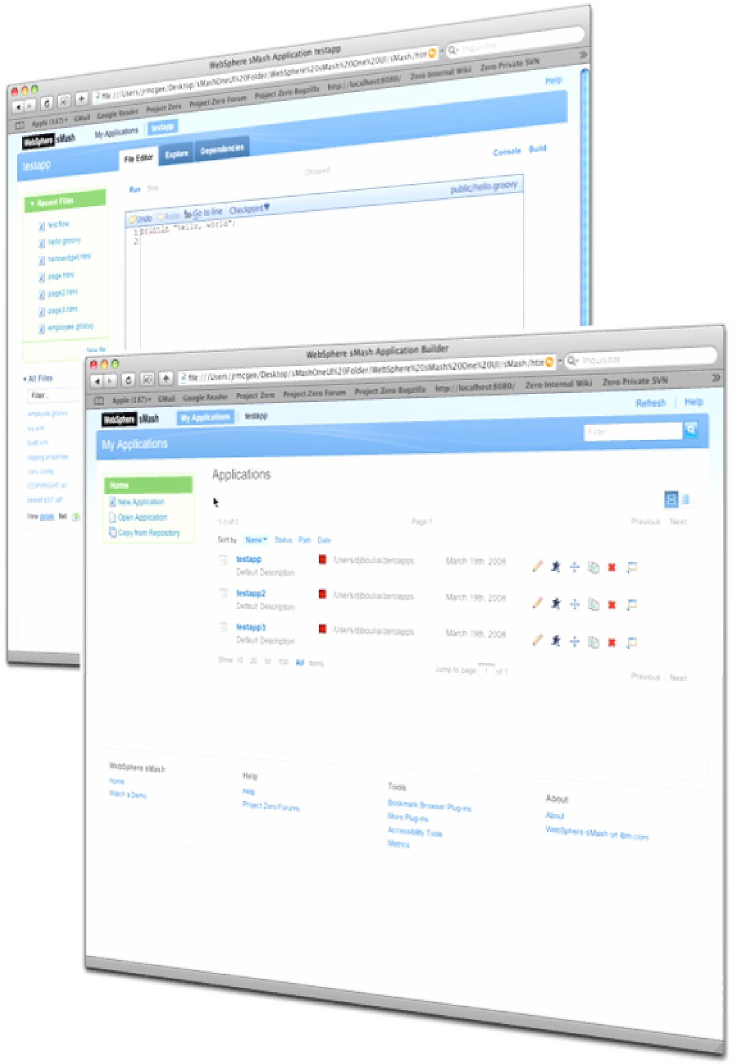
Dynamic Scripting and Templates

Effortless creation of RESTful Services and Data Feeds (RSS, ATOM)

Simple Event-based execution environment

State externalized into a shared memory space (Global Context)

Repository of pre-built Services and Libraries provides useful building blocks



```
print "Hello, world";
```

DYNAMIC SCRIPTING

WebSphere sMash is a dynamic scripting platform

Application Logic is created in one of two scripting languages

- Groovy (for people that prefer Java)
- PHP (for the 3 Million existing PHP programmers)

Java is positioned as the “system” language

- Mostly used to implement system extensions and application libraries
- Entire applications can be written in Java, if desired
  - Requires more configuration

APPLICATION CENTRIC RUNTIME

WebSphere sMash is an application-centric runtime

- You create an application and run it
- You do not package an application and deploy it to a multi-application server
- Each application runs in its own process (JVM)
- Runtime is designed to be short lived
  - Update recycles after idle timeout or max number of requests

WebSphere sMash is a full stack runtime

- Everything needed to run the application is built in
  - Including the HTTP stack
- No external proxy or web server is required
- Does not deploy as a WAR file inside another JEE container
- An external proxy is used for clustering and multi-app routing

PHP SUPPORT

- Gartner predict that within 5 years 60% of the 5.5 million PHP programmers will work in corporate IT. (Up from 13% of 3M today).
- The PHP runtime is built on top of a standard JVM
  - Supports use of many PHP Extensions
    - XAPI-C interface allows C-based extensions
    - XAPI-J interface allows Java based extensions
  - Supports bridging between Java and PHP
  - Currently supports a subset of PHP
    - The goal is maximum re-use of existing PHP scripts
- PHP runtime provided directly by WebSphere sMash, not php.net
- The goal of PHP support is about unleashing the 3 Million PHP programmers together with the vast library of existing PHP script code and extensions and bringing it to the sMash programming model
- A number of popular PHP application now run on the sMash PHP runtime, including
  - **phpBB**
  - **SugarCRM**

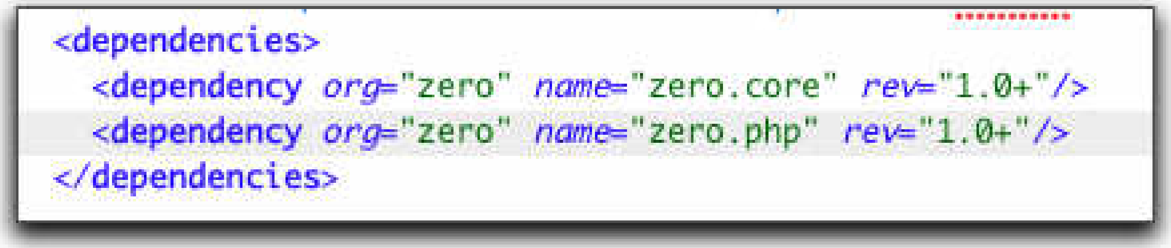
MODULAR ARCHITECTURE

WebSphere sMash applications are based on a very small core

- 5.4 MBytes (includes Groovy).
  - PHP adds additional 14.5 Mbytes
    - Contains 3 platforms currently
- Core provides all of the framework and runtime support, including HTTP transport

Additional features provided in downloadable modules

- Applications declare a dependency on desired features (using Ivy)
- A package management system manages your dependencies, including:
  - The ability to share dependencies on a machine
  - The ability to demand load missing dependencies from the network
  - The ability to manage updates to dependencies that you are using



```
<dependencies>
  <dependency org="zero" name="zero.core" rev="1.0+"/>
  <dependency org="zero" name="zero.php" rev="1.0+"/>
</dependencies>
```

SIMPLE DEPLOYMENT

Essentially the deployment is Zip and Copy

No machine specific information bound into the application

Default mode is shared dependencies

- Application dependencies are load for the deployment machines local repository and pulled off the network if needed

Standalone mode is supported as well

- All application dependencies are included in the ZIP and nothing is needed on the target machine except a JVM

Provides a packaging command to simplify the creation of the ZIP file for deployment

- zero package for shared mode
- zero package standalone for standalone mode

RUNTIME CHARACTERISTICS

Instant On

- Application Available for Service in less than 1 sec
  - 0.672 seconds on a MacBook Pro
- Application JVM starts in about 1 second
  - 1.3 seconds on a MacBook Pro

Clean

- Graceful recovery, isolation, tolerates “bad” code
- Short lived processes
  - Runs for fixed number of requests or idle timeout then restarts
  - No state lost on restart

Cheap

- Cost effective to run in small and large quantities
- Idle Application Footprint ~380 Kbytes
- Running Application JVM ~28 Mbytes TODAY

Supported on “stock” JVM

- IBM, Sun, Mac, etc - Any JSE 5 or 6 JVM

BUILT-IN DEVELOPMENT TOOLING

WEBSPERE SMASH LETS DEVELOPERS BUILD APPLICATION DIRECTLY ON THE WEB

Browser-Based Development IDE

Built as a sMash application

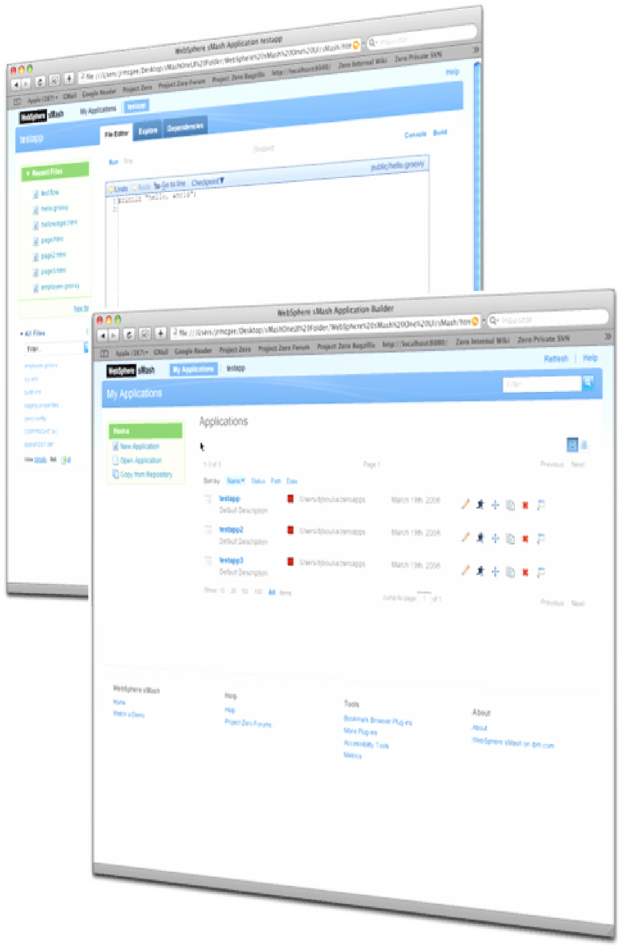
Provides full development lifecycle for Zero applications

- Create, Edit, Test

Provides Visual Editors for Activities and Web Page construction

- Including a DOJO-enabled page editor

Basic Eclipse-based tooling also available if required


WebSphere sMash Application testapp
WebSphere sMash Application Builder
My Applications
Applications
testapp
testapp2
testapp3

WEBSPHERE SMASH ACTIVITIES

LETS DEVELOPERS VISUALLY "MASH-UP" SERVICES AND FEEDS

Assemble-style Development

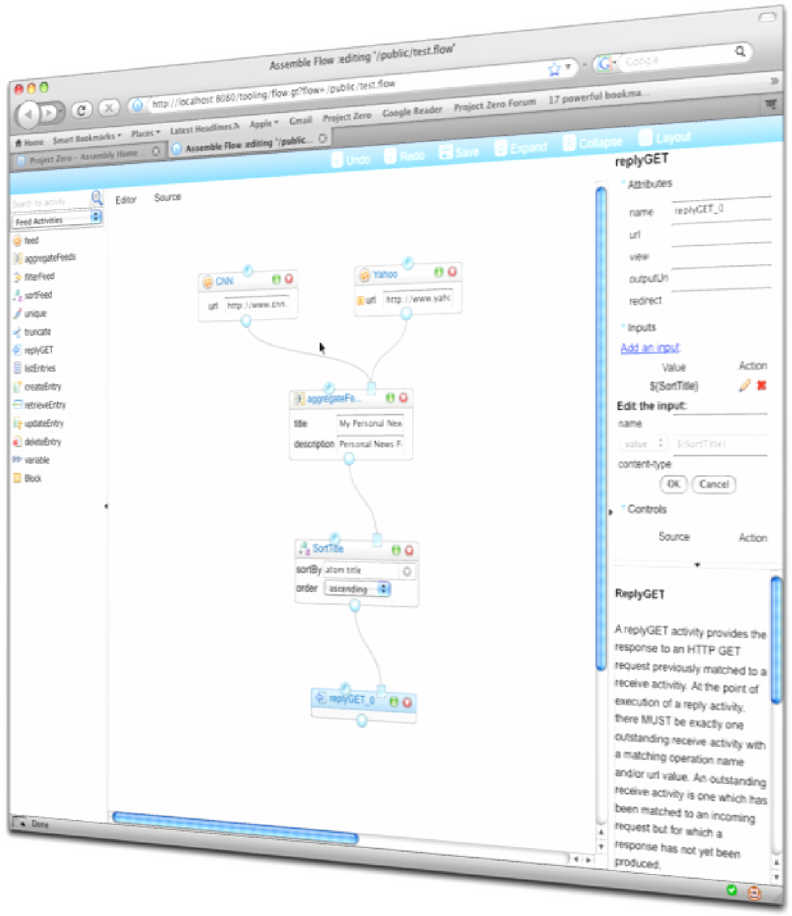
Compose applications by "wiring" together REST services

Visually or programmatically combine existing feeds and services that enrich, sort, and filter data in a pipeline

Configure templates to alter pipeline routes, log events along the pipeline

Numerous built-in activities, including

Get Feed, Call Service, Aggregate, Sort, Transform, Filter, Send Mail, XSLT, Conditionals, Loops

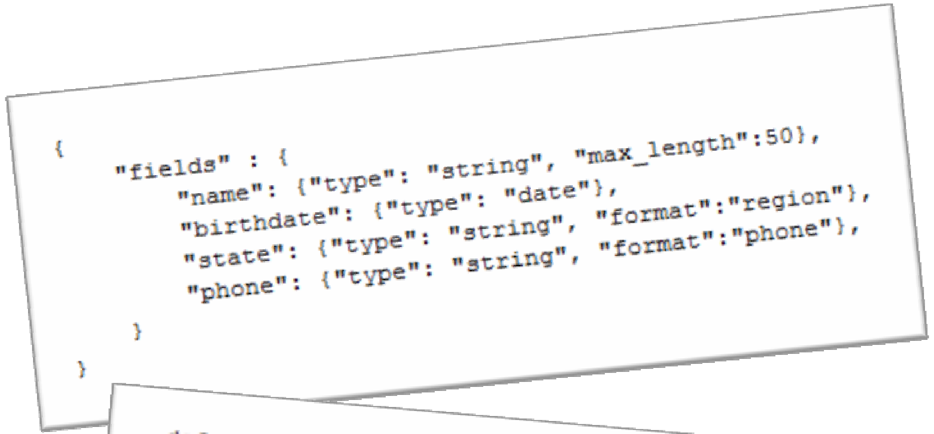


ReplyGET

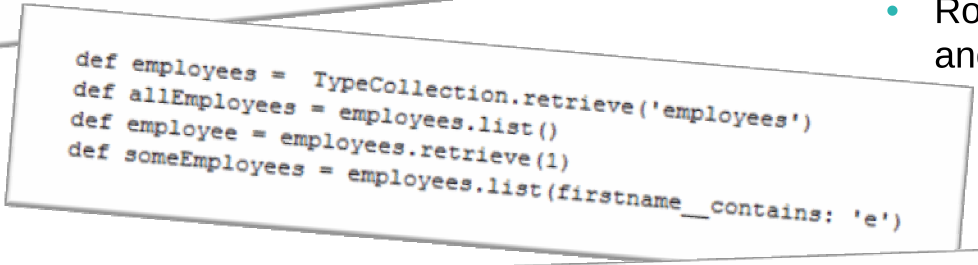
A replyGET activity provides the response to an HTTP GET request previously matched to a receive activity. At the point of execution of a reply activity, there MUST be exactly one outstanding receive activity with a matching operation name and/or url value. An outstanding receive activity is one which has been matched to an incoming request but for which a response has not yet been produced.

RAPIDLY EXPOSE DATA RESTFULLY

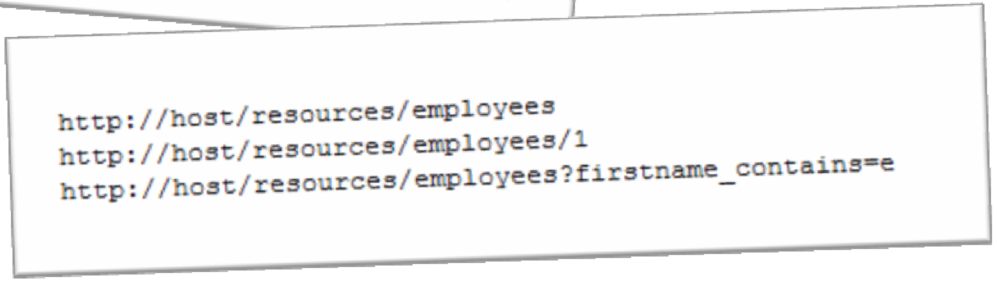
ZERO RESOURCE MODEL ENABLES DEVELOPERS WITH A SIMPLE PROGRAMMATIC AND HTTP DATA API



```
{
    "fields" : {
        "name": {"type": "string", "max_length":50},
        "birthdate": {"type": "date"},
        "state": {"type": "string", "format":"region"},
        "phone": {"type": "string", "format":"phone"},
    }
}
```



```
def employees =  TypeCollection.retrieve('employees')
def allEmployees = employees.list()
def employee = employees.retrieve(1)
def someEmployees = employees.list(firstname__contains: 'e')
```



```
http://host/resources/employees
http://host/resources/employees/1
http://host/resources/employees?firstname_contains=e
```

Model application data

- Constrained set of APIs encourage a RESTful application architecture
- Data model that maps well into Atom feeds and JSON formats
- Robust framework for persistence, validation, and serialization

AVAILABLE MODULES

- There are approximately 65 modules available currently
- Modules provide function in many categories
  - Data Formats (JSON, ATOM, RSS, XML)
  - Data Access
  - Resource Modeling
  - Security / Content Filtering
  - Activity Flows
  - Services
    - Amazon ECS, Flickr, Weather, etc
  - Utilities (such as HTML parsing)
  - Management Tools
  - Development Tooling
  - Reliable Transport Engine for Messaging Interactions

COMMUNITY DRIVEN COMMERCIAL DEVELOPMENT

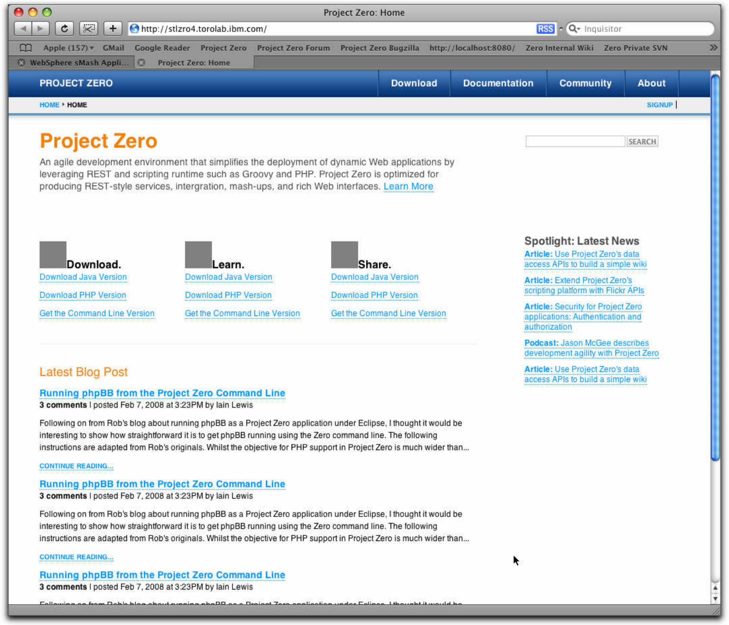
Evolve the core platform based on developer feedback

Commercial development using a transparent development process

Enabled via an external web site providing:

- *A focal point for all sMash development activities*
  - Expose the IBM development process to the external developer community
  - All design decisions are discussed and communicated via external forums
  - Registered users can post comments and feedback to the forums
- *Frictionless download of latest code and documentation*
  - Registration not required for binary downloads
  - Latest builds immediately available to developers
  - Source code can be viewed by registered users

http://www.projectzero.org


Project Zero: Home
http://stlzro4.torolab.ibm.com/
RSS
Inquisitor
Apple (157) GMail Google Reader Project Zero Project Zero Forum Project Zero Bugzilla http://localhost:8080/ Zero Internal Wiki Zero Private SVN
WebSphere sMash Appli... Project Zero: Home
PROJECT ZERO
Download
Documentation
Community
About
HOME › HOME
SIGNUP
SEARCH

Project Zero

An agile development environment that simplifies the deployment of dynamic Web applications by leveraging REST and scripting runtime such as Groovy and PHP. Project Zero is optimized for producing REST-style services, intergration, mash-ups, and rich Web interfaces. Learn More

Download.
Download Java Version
Download PHP Version
Get the Command Line Version

Learn.
Download Java Version
Download PHP Version
Get the Command Line Version

Share.
Download Java Version
Download PHP Version
Get the Command Line Version

Spotlight: Latest News
Article: Use Project Zero's data access APIs to build a simple wiki
Article: Extend Project Zero's scripting platform with Flickr APIs
Article: Security for Project Zero applications: Authentication and authorization
Podcast: Jason McGee describes development agility with Project Zero
Article: Use Project Zero's data access APIs to build a simple wiki

Latest Blog Post

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

Following on from Rob's blog about running phpBB as a Project Zero application under Eclipse, I thought it would be interesting to show how straightforward it is to get phpBB running using the Zero command line. The following instructions are adapted from Rob's originals. Whilst the objective for PHP support in Project Zero is much wider than...

CONTINUE READING...

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

Following on from Rob's blog about running phpBB as a Project Zero application under Eclipse, I thought it would be interesting to show how straightforward it is to get phpBB running using the Zero command line. The following instructions are adapted from Rob's originals. Whilst the objective for PHP support in Project Zero is much wider than...

CONTINUE READING...

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

PROJECTZERO.ORG
LOWERING THE BARRIERS TO COMMUNITY ACCESS

Anonymous Visitors can…

Browse the site
View Wiki content
Read Forums
Search the Bug Database
Read Blogs
Download Binary Drivers*

Focused on easy access

- Internet web site
- Free access to the platform

Registered Users can…

Post to the Forum
Submit Bug Reports
Submit Feature Requests
Comment on Blog Posts
Access Source Code*

Focused on feedback

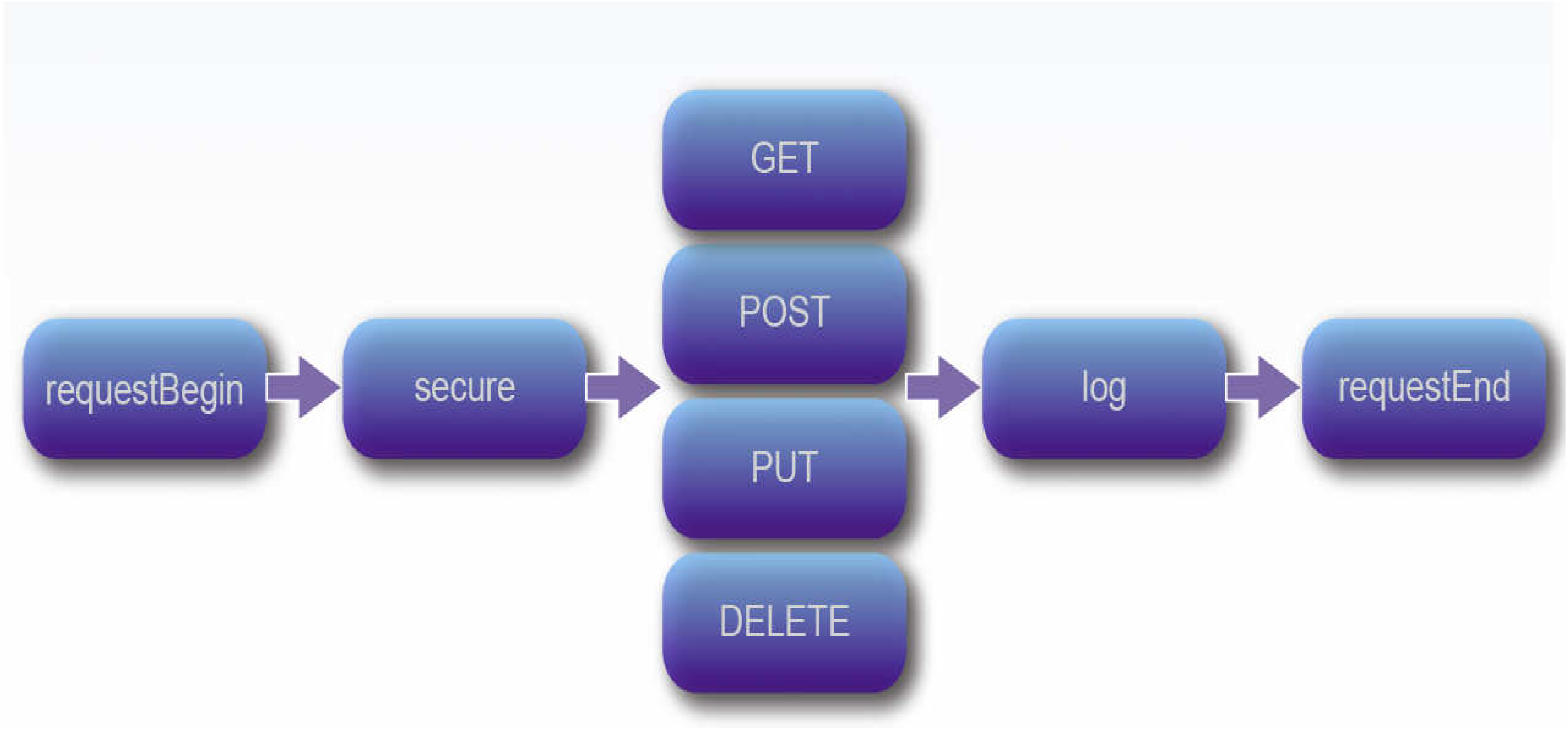
- Simple, free registration process

* Requires acceptance of an IBM license agreement

Core Programming Model

Events

- All behavior in the system is modeled as a set of event
  - Applications are built by handling these events and providing desired behavior
  - Similar to AJAX model or classic UI programming



Event Handlers

All handlers are stateless

Can be implemented in Groovy, PHP, and Java

Groovy

```
println "Hello World"
```

```
def onGET()
{
   println "Hello World"
}
```

PHP

```
function onGET()
{
   echo "Hello World";
}
```

Java

```
public void onGET()
{
   PrintWriter writer = (PrintWriter)zget("/request/writer");
   writer.println("Hello World");
}
```

```
/config/handlers += {
   "events" : "GET",
   "handler" : "HelloWorld.class",
   "conditions" : ["/request/path matches /hello"]
}
```

Global Context – State Management

The Global Context (GC) provides access to and management of all application state

- Conceptually a map of data

Externalizes all state from the application logic

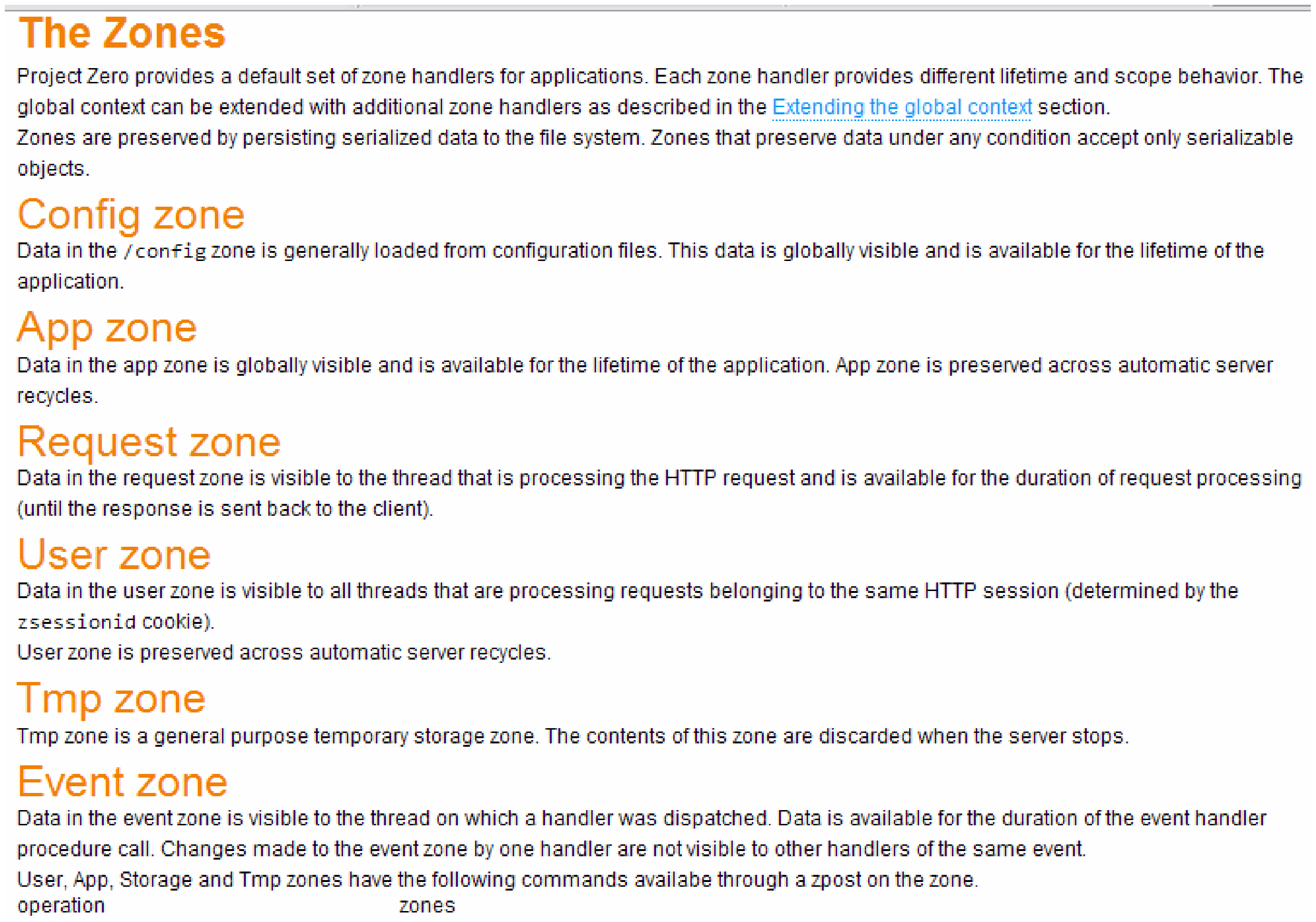
- Enables the restart ability of the JVM without data loss
- Enables clustering and scaling to be added transparently

Simplifies and unifies access to application state and data structures and simplifies state passing within the application

Contains information provided by both the runtime (such as request parameters) and by the application

Global Context Zone

Divided into zones representing different data lifecycles



The Zones

Project Zero provides a default set of zone handlers for applications. Each zone handler provides different lifetime and scope behavior. The global context can be extended with additional zone handlers as described in the Extending the global context section.

Zones are preserved by persisting serialized data to the file system. Zones that preserve data under any condition accept only serializable objects.

Config zone

Data in the `/config` zone is generally loaded from configuration files. This data is globally visible and is available for the lifetime of the application.

App zone

Data in the app zone is globally visible and is available for the lifetime of the application. App zone is preserved across automatic server recycles.

Request zone

Data in the request zone is visible to the thread that is processing the HTTP request and is available for the duration of request processing (until the response is sent back to the client).

User zone

Data in the user zone is visible to all threads that are processing requests belonging to the same HTTP session (determined by the `zsessionid` cookie).

User zone is preserved across automatic server recycles.

Tmp zone

Tmp zone is a general purpose temporary storage zone. The contents of this zone are discarded when the server stops.

Event zone

Data in the event zone is visible to the thread on which a handler was dispatched. Data is available for the duration of the event handler procedure call. Changes made to the event zone by one handler are not visible to other handlers of the same event.

User, App, Storage and Tmp zones have the following commands availabe through a zpost on the zone.

operation	zones

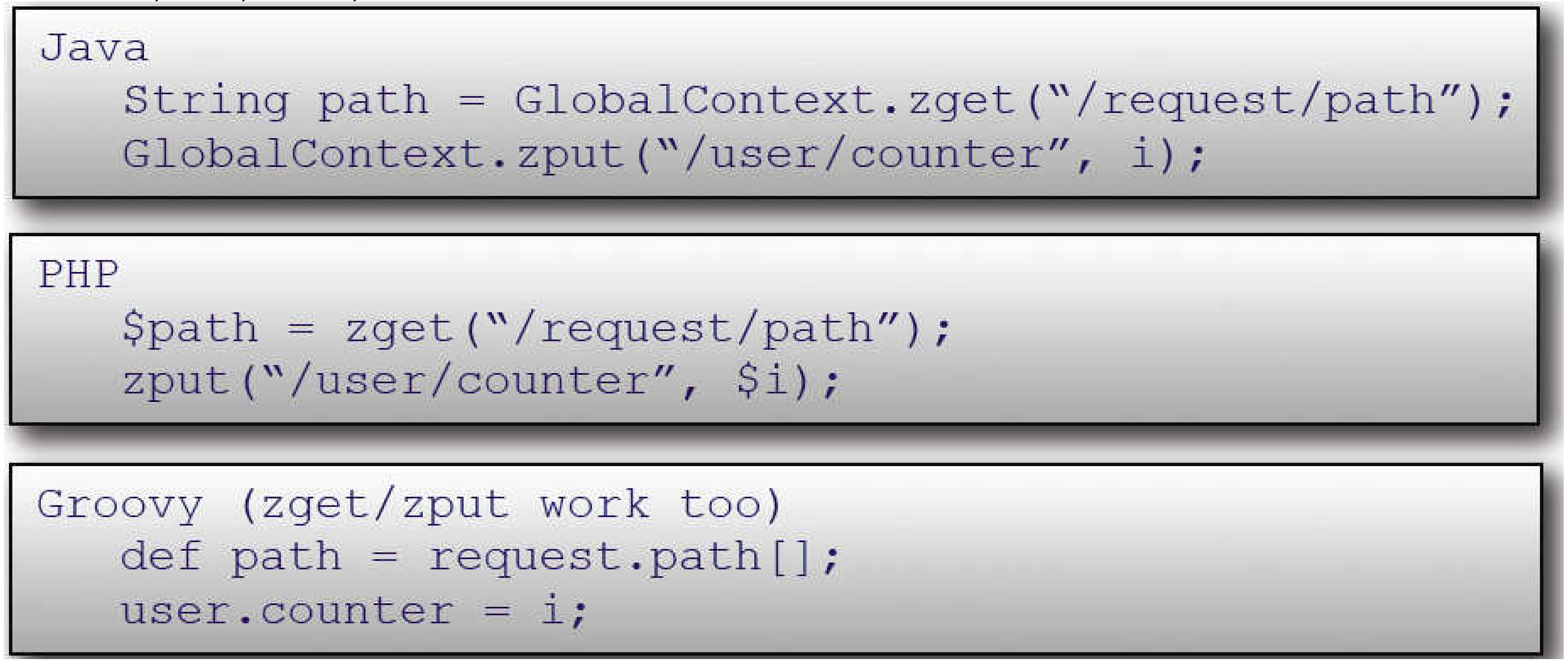
Accessing the Global Context

Data is organized by a URI structure

- First part of URI is always the Zone name
  - /app, /user, /request, /config, /event, /client

Access is modeled after REST

- GET, PUT, POST, DELETE



```
Java
    String path = GlobalContext.zget("/request/path");
    GlobalContext.zput("/user/counter", i);
```

```
PHP
    $path = zget("/request/path");
    zput("/user/counter", $i);
```

```
Groovy (zget/zput work too)
    def path = request.path[];
    user.counter = i;
```

Value Pathing

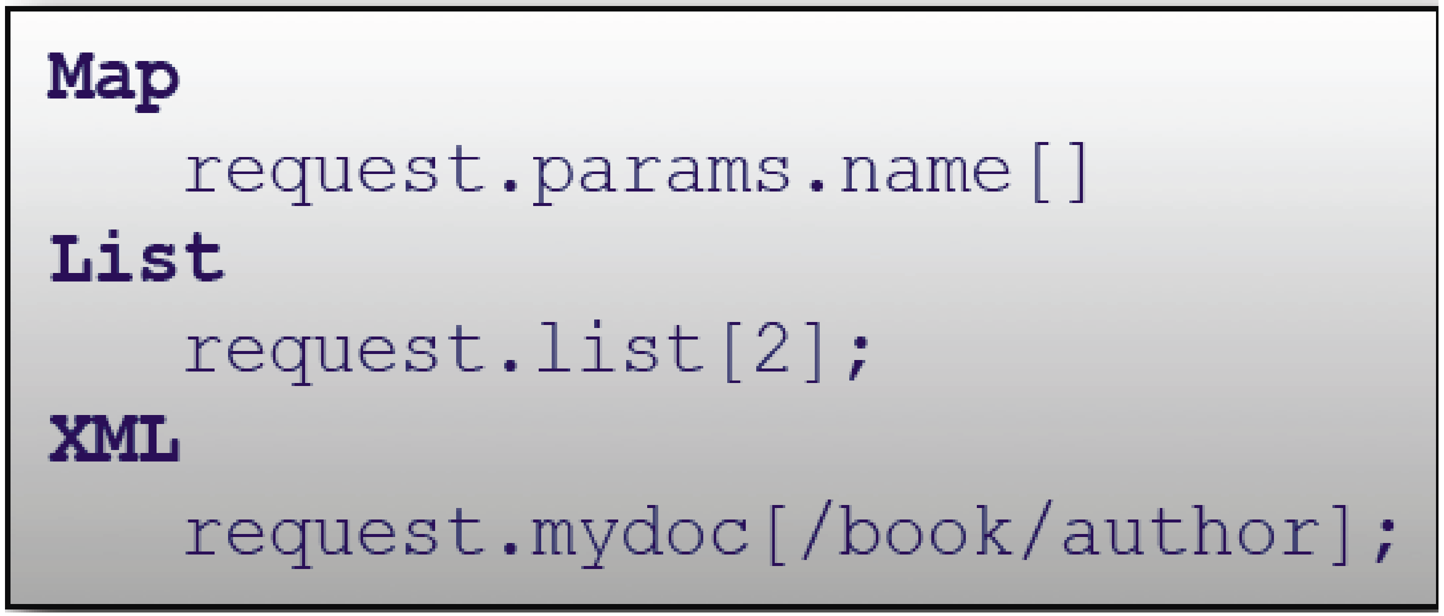
The GC provides simplified access to certain data structures

Called ***Value Pathing***

Understands

Maps, List, Objects, XML, JSON

□Allows read and write access to internals of the structure through the GC address



Map

```
    request.params.name[]
```

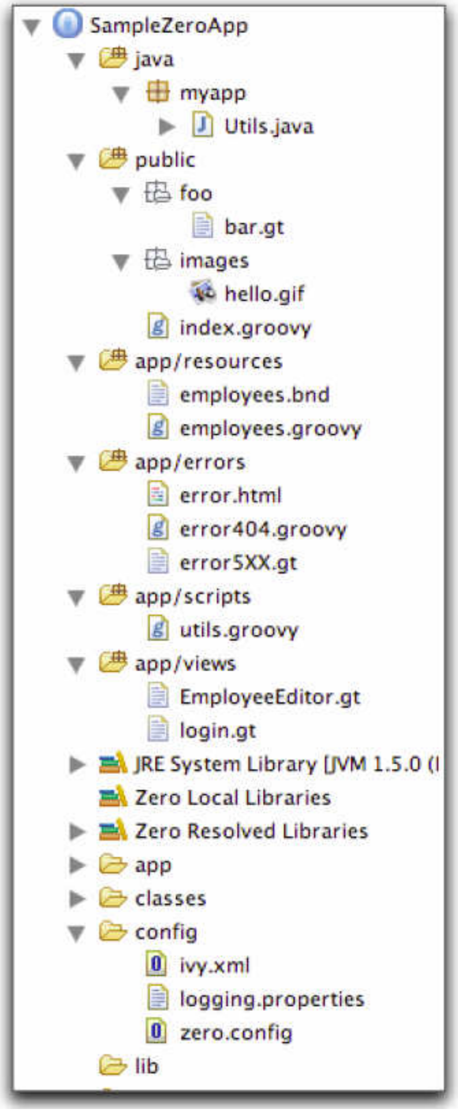
List

```
    request.list[2];
```

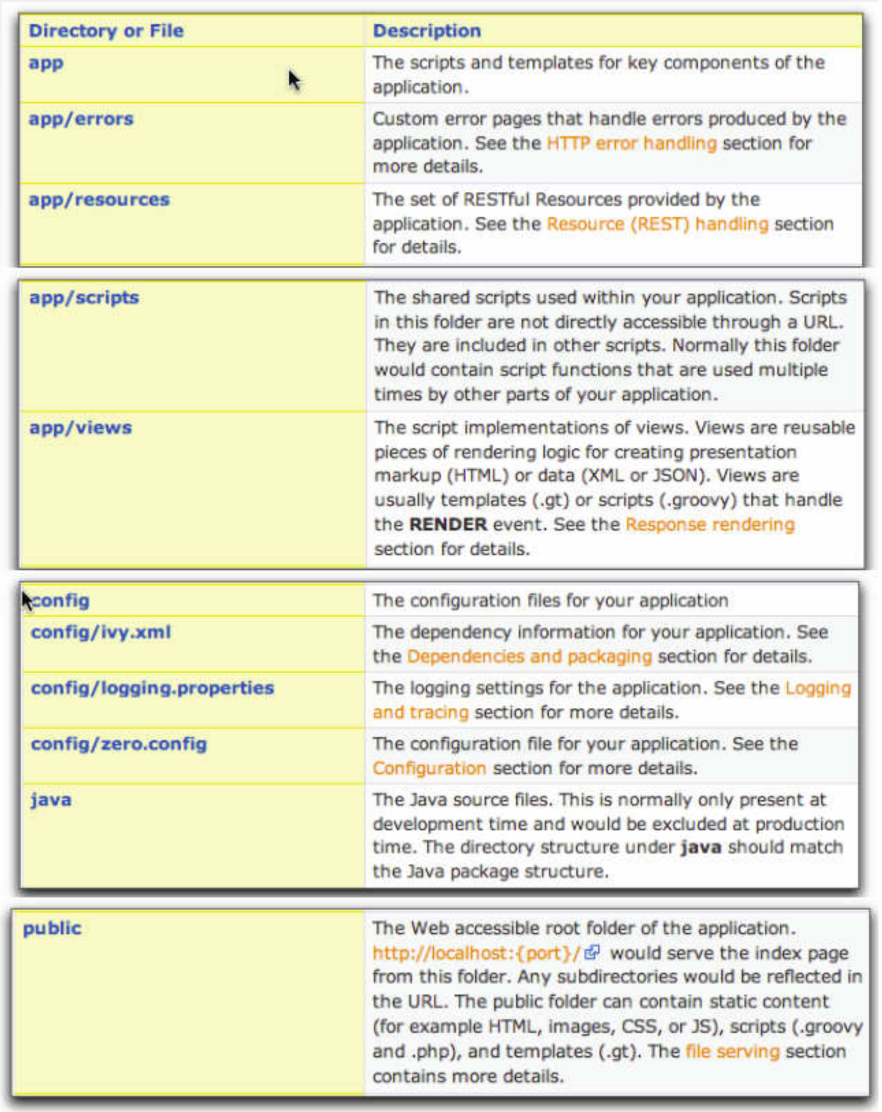
XML

```
    request.mydoc[/book/author];
```

Application Directory Layout



```
▼ SampleZeroApp
  ▼ java
    ▼ myapp
      ▶ Utils.java
  ▼ public
    ▼ foo
        bar.gt
    ▼ images
        hello.gif
      index.groovy
  ▼ app/resources
      employees.bnd
      employees.groovy
  ▼ app/errors
      error.html
      error404.groovy
      error5XX.gt
  ▼ app/scripts
      utils.groovy
  ▼ app/views
      EmployeeEditor.gt
      login.gt
  ▶ JRE System Library [JVM 1.5.0 (I
    Zero Local Libraries
  ▶ Zero Resolved Libraries
  ▶ app
  ▶ classes
  ▼ config
      ivy.xml
      logging.properties
      zero.config
    lib
```

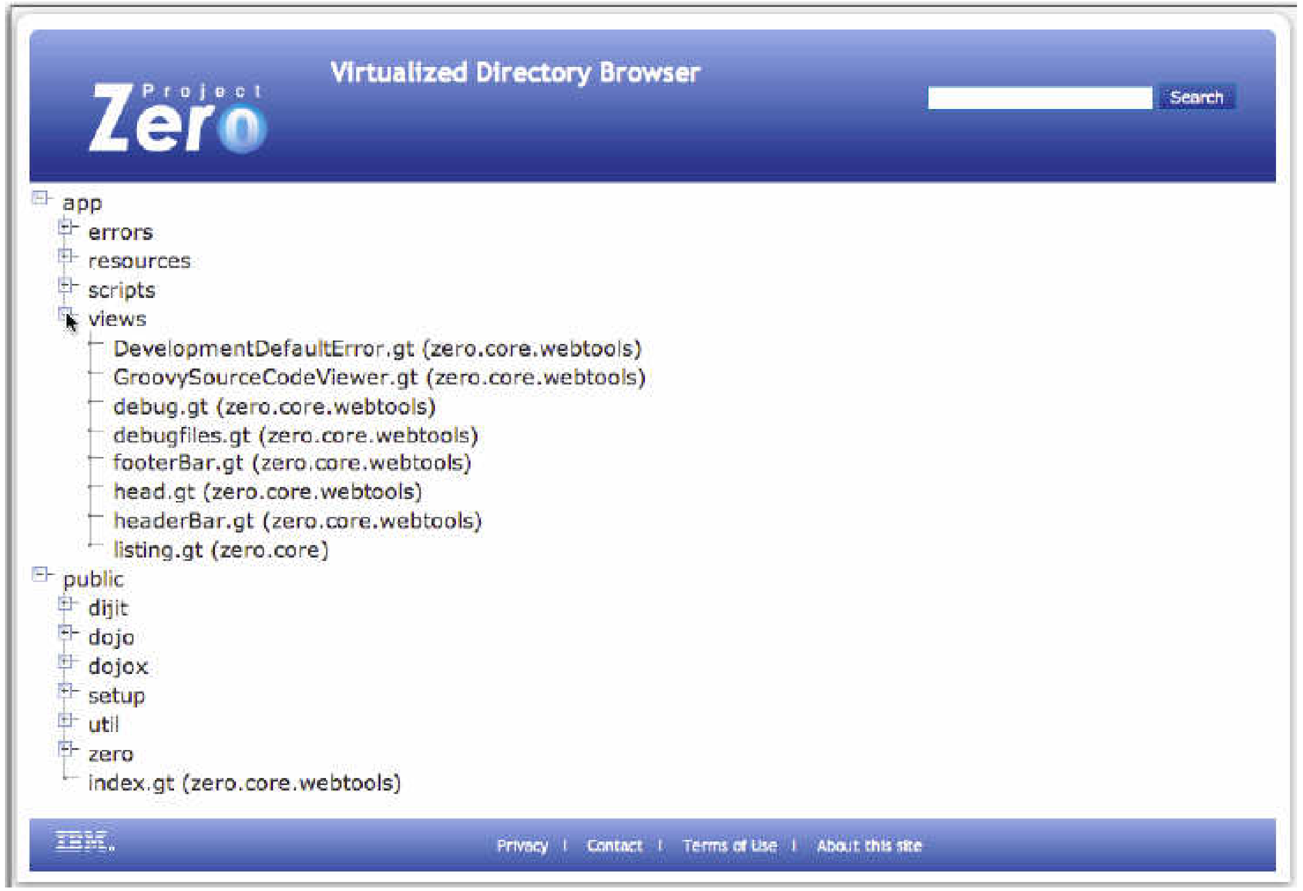


Directory or File	Description
app	The scripts and templates for key components of the application.
app/errors	Custom error pages that handle errors produced by the application. See the HTTP error handling section for more details.
app/resources	The set of RESTful Resources provided by the application. See the Resource (REST) handling section for details.
app/scripts	The shared scripts used within your application. Scripts in this folder are not directly accessible through a URL. They are included in other scripts. Normally this folder would contain script functions that are used multiple times by other parts of your application.
app/views	The script implementations of views. Views are reusable pieces of rendering logic for creating presentation markup (HTML) or data (XML or JSON). Views are usually templates (.gt) or scripts (.groovy) that handle the **RENDER** event. See the Response rendering section for details.
config	The configuration files for your application
config/ivy.xml	The dependency information for your application. See the Dependencies and packaging section for details.
config/logging.properties	The logging settings for the application. See the Logging and tracing section for more details.
config/zero.config	The configuration file for your application. See the Configuration section for more details.
java	The Java source files. This is normally only present at development time and would be excluded at production time. The directory structure under **java** should match the Java package structure.
public	The Web accessible root folder of the application. http://localhost:{port}/ would serve the index page from this folder. Any subdirectories would be reflected in the URL. The public folder can contain static content (for example HTML, images, CSS, or JS), scripts (.groovy and .php), and templates (.gt). The file serving section contains more details.

Virtualized Directory

Project Zero provides seamless integration of directories across an application and its dependencies, while maintaining each as separate entities.

All artifacts are searched within both the application and its declared dependencies


Project Zero
Virtualized Directory Browser
Search
Privacy | Contact | Terms of Use | About this site

```
app
  errors
  resources
  scripts
  views
    DevelopmentDefaultError.gt (zero.core.webtools)
    GroovySourceCodeViewer.gt (zero.core.webtools)
    debug.gt (zero.core.webtools)
    debugfiles.gt (zero.core.webtools)
    footerBar.gt (zero.core.webtools)
    head.gt (zero.core.webtools)
    headerBar.gt (zero.core.webtools)
    listing.gt (zero.core)
public
  dijit
  dojo
  dojox
  setup
  util
  zero
  index.gt (zero.core.webtools)
```

Configuration

Zero configuration file: zero.config

The config/zero.config file is processed at the start of a Zero application.

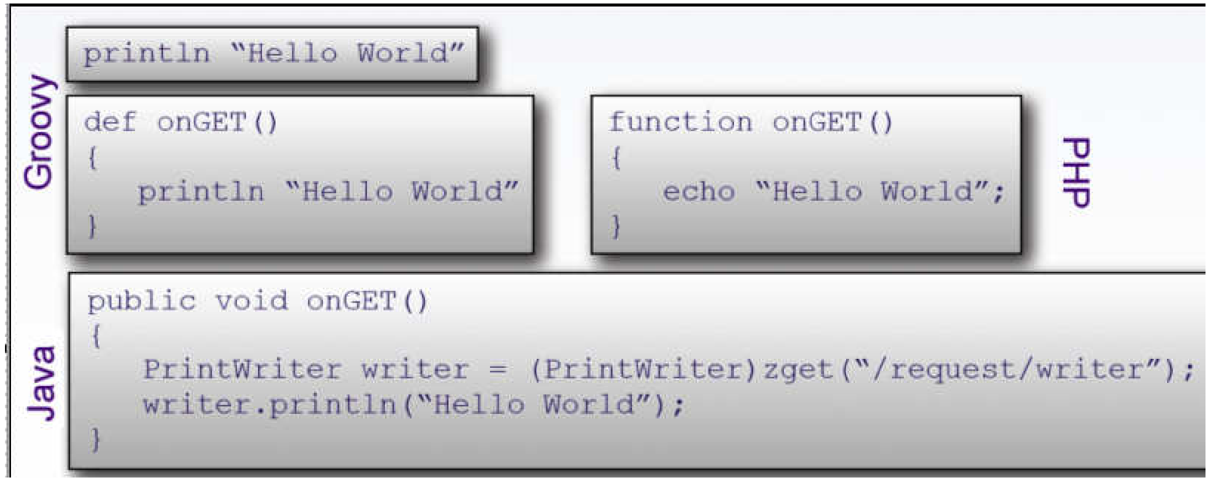
The content of a config/zero.config file is organized into "stanzas" of related key/value pairs. Stanzas are associated with directives, such as "store to the Global Context" and "include another configuration file."

```
# Value set
/config/http/port = 8080

# List set
/config/resources/defaultExtensions = [".groovy"]

# List append
/config/bindings/.groovy +=
["zero.core.groovysupport.bindings.DefaultBindings"]

# Map set
/config/test/map = { "a" : "b", "c" : "d" }

# Map append
/config/test/mapappend += { "a" : "b", "c" : "d" }
/config/test/mapappend += { "x" : "y", "w" : "z" }

# Event handler
/config/handlers += {
  "events" : "GET",
  "handler" : "custom.Handler.class"
}

# Value reference (insert value read at config-load time)
/config/property/myPrefix = "/foo/bar"
/config/test/value = "${/config/property/myPrefix}/bat"

# Variable set/value reference
myPrefix = "/foo/bar"
/config/test/value = "${myPrefix}/bat"

# Include
@include "${/config/dependencies/zero.core}/config/security/
form.config"
{ "formLoginPage" : "/login" }
```

Rendering

Direct

Indirect

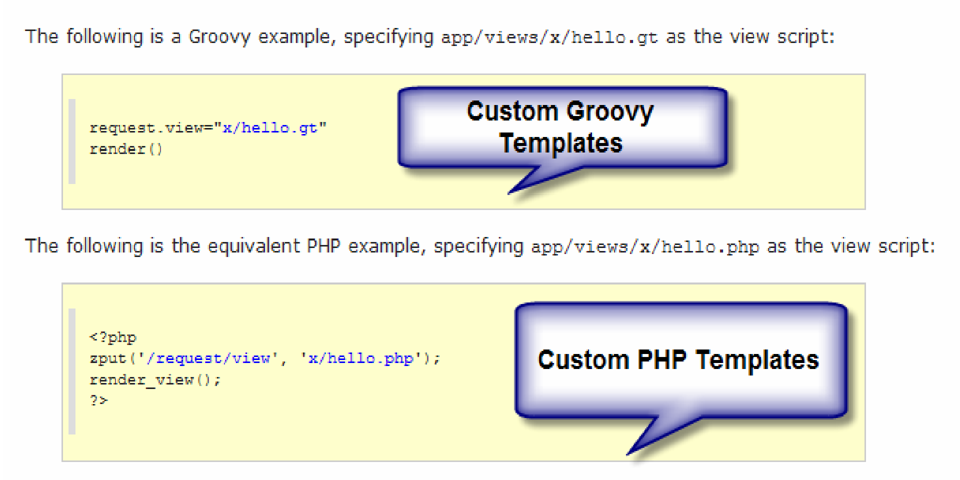


```
println "Hello World"
```

```
def onGET()
{
   println "Hello World"
}
```

```
function onGET()
{
   echo "Hello World";
}
```

```
public void onGET()
{
   PrintWriter writer = (PrintWriter)zget("/request/writer");
   writer.println("Hello World");
}
```

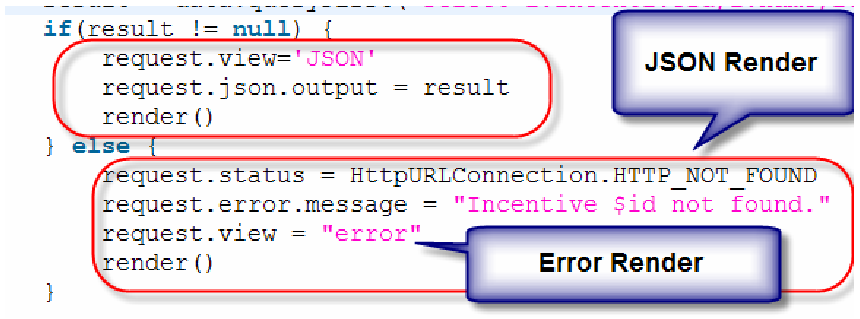


The following is a Groovy example, specifying `app/views/x/hello.gt` as the view script:

```
request.view="x/hello.gt"
render()
```

The following is the equivalent PHP example, specifying `app/views/x/hello.php` as the view script:

```
<?php
zput('/request/view', 'x/hello.php');
render_view();
?>
```



```
if(result != null) {
    request.view='JSON'
    request.json.output = result
    render()
} else {
    request.status = HttpURLConnection.HTTP_NOT_FOUND
    request.error.message = "Incentive $id not found."
    request.view = "error"
    render()
}
```

REST

What is REST?

Representational State Transfer

Architectural model on which the World Wide Web is based

Principles of REST

- Resource-centric approach
- All relevant resources are addressable via URIs
- Uniform access via HTTP – GET, POST, PUT, DELETE
- Content-type negotiation allows retrieving alternative representations from same URI

REST-style services

- are easy to access from code running in web browsers, any other client or servers
- can serve multiple representations of the same resource

Accessibility for Developers

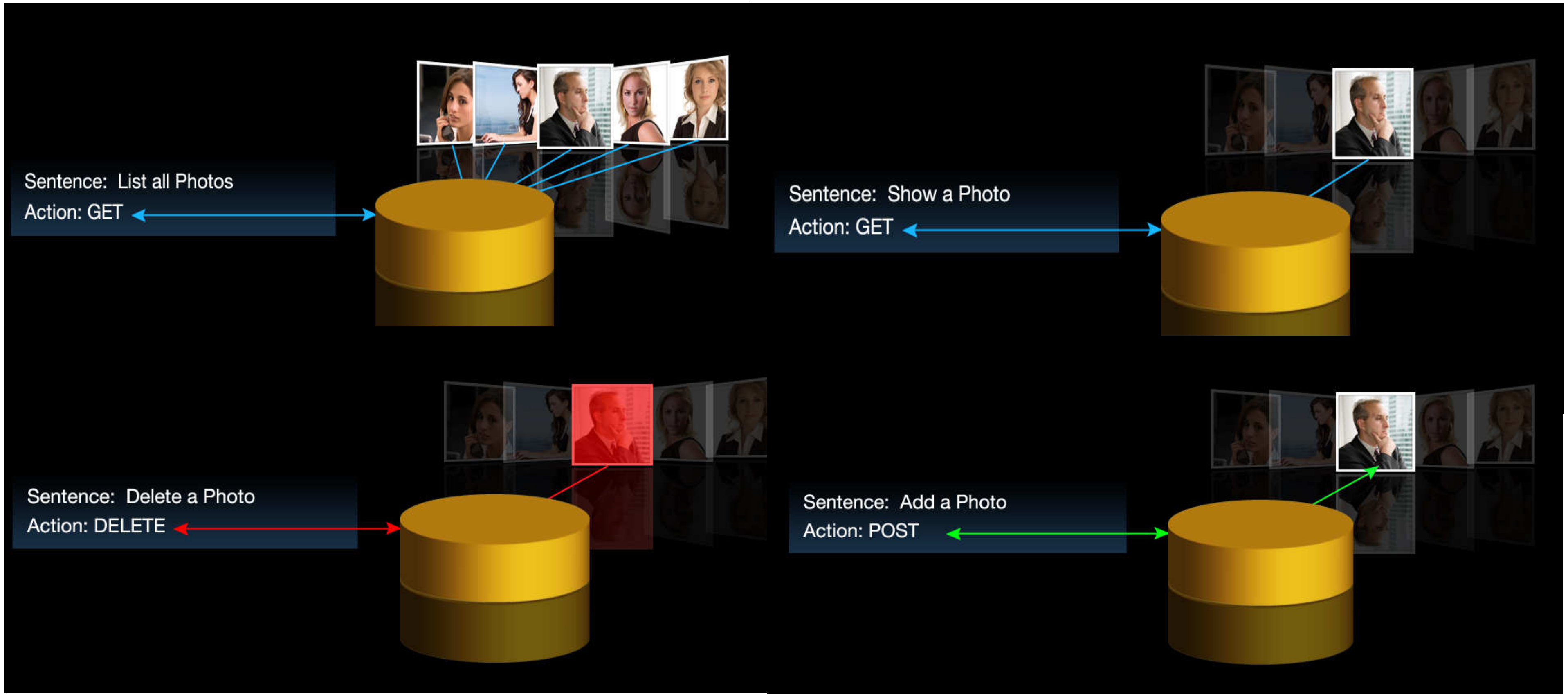
Simply exposing services from the enterprise as URLs and Feeds

A RESTful Web service is formed like a sentence:

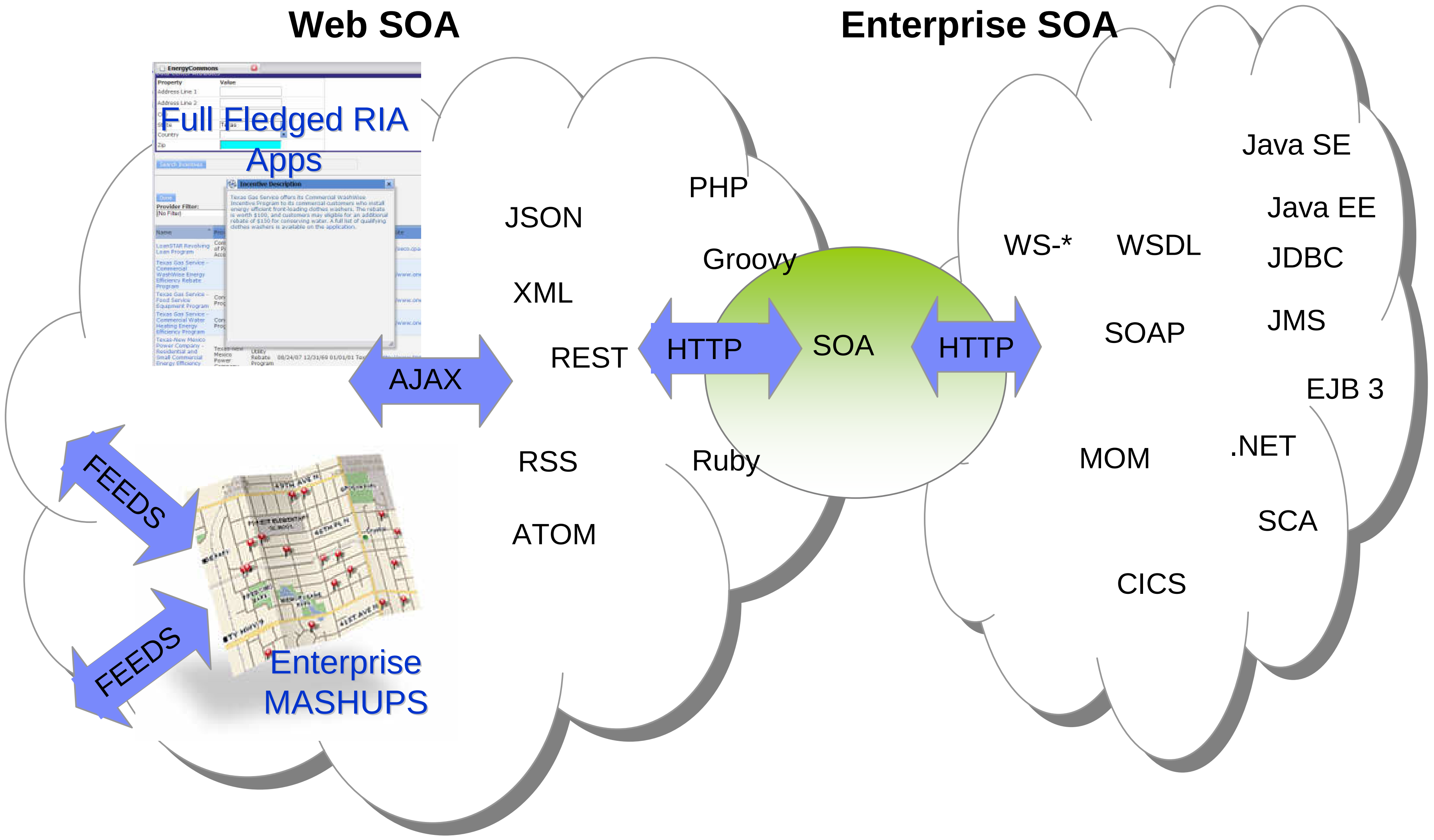
Verb = HTTP Action (GET, POST, PUT, DELETE)

Noun = the URI of the Service (the document)

Adjective = MIME type of the resulting document



Bridging Web SOA and Enterprise SOA



WebSphere sMash and REST

Resources on the Web

What are the URIs?

Which methods are supported at each URI?

What formats?

Resource	**URI**	**Method**	**Representation**	**Description**
Employee list	/resources/employee	GET	JSON (list)	List
		POST	JSON (employee)	Create
Employee	/resources/employee/{id}	GET	JSON (employee)	Retrieve
		PUT	JSON (employee)	Update
		DELETE		Delete

Representations

Employee

```
{
  "first_name“ : "Alice",
  "last_name“  : "Rogers",
  "location"   : "Seattle"
}
```

List of employees

```
[{
  "first_name“ : "Alice",
  "last_name“  : "Rogers",
  "location"   : "Seattle"
 },
 {
  "first_name“ : "Bill”,
  "last_name"  : "Stevens",
  "location"   : "Seattle“
}]
```

Resource Handlers in Zero

Basic event handlers for /resources/*

URI pattern	Method	Event	Description
/resources/collection	GET	list	List of all members
	POST	create	Create member
/resources/collection/{id}	GET	retrieve	Retrieve one member
	PUT	update	Replace member
	DELETE	delete	Delete member

Resource Handlers Example

app/resources/employee.groovy

```
def onList() {
  try {
    // Get configured DataManager for data access
    def data = zero.data.groovy.Manager.create('employee_db')

    // Retrieve employee records via Data Zero
    def result = data.queryArray('SELECT * FROM employees')
    request.view = 'JSON'
    request.json.output = result
    render()

  } catch (Exception e){
    if (e.getCause() instanceof java.sql.SQLException) {
      request.status = HttpURLConnection.HTTP_INTERNAL_ERROR
      request.view = 'error'
      request.error.message = 'The db may not have been initialized.'
      render()
    }
  }
} def onCreate() { ...
```

Resource Handlers Example

app/resources/employee.groovy (continued)

```
def onCreate() {
  // Convert entity to JSON object
  def emp = zero.json.Json.decode(request.input[])

  // Get configured DataManager for data access
  def data = zero.data.groovy.Manager.create('employee_db')

  // Insert employee record via Data Zero APIs
  data.update("""
    INSERT INTO employees
      (username, firstname, lastname, location, phonenumber)
    VALUES ($emp.username, $emp.firstname, $emp.lastname,
            $emp.location, $emp.phonenumber)
""")

  // Set a Location header with URI to the new record
  locationUri = getRequestedUri(false) + '/' + emp['username']
  request.headers.out.Location = locationUri
  request.status = HttpURLConnection.HTTP_NO_CONTENT
}
```

Renderers

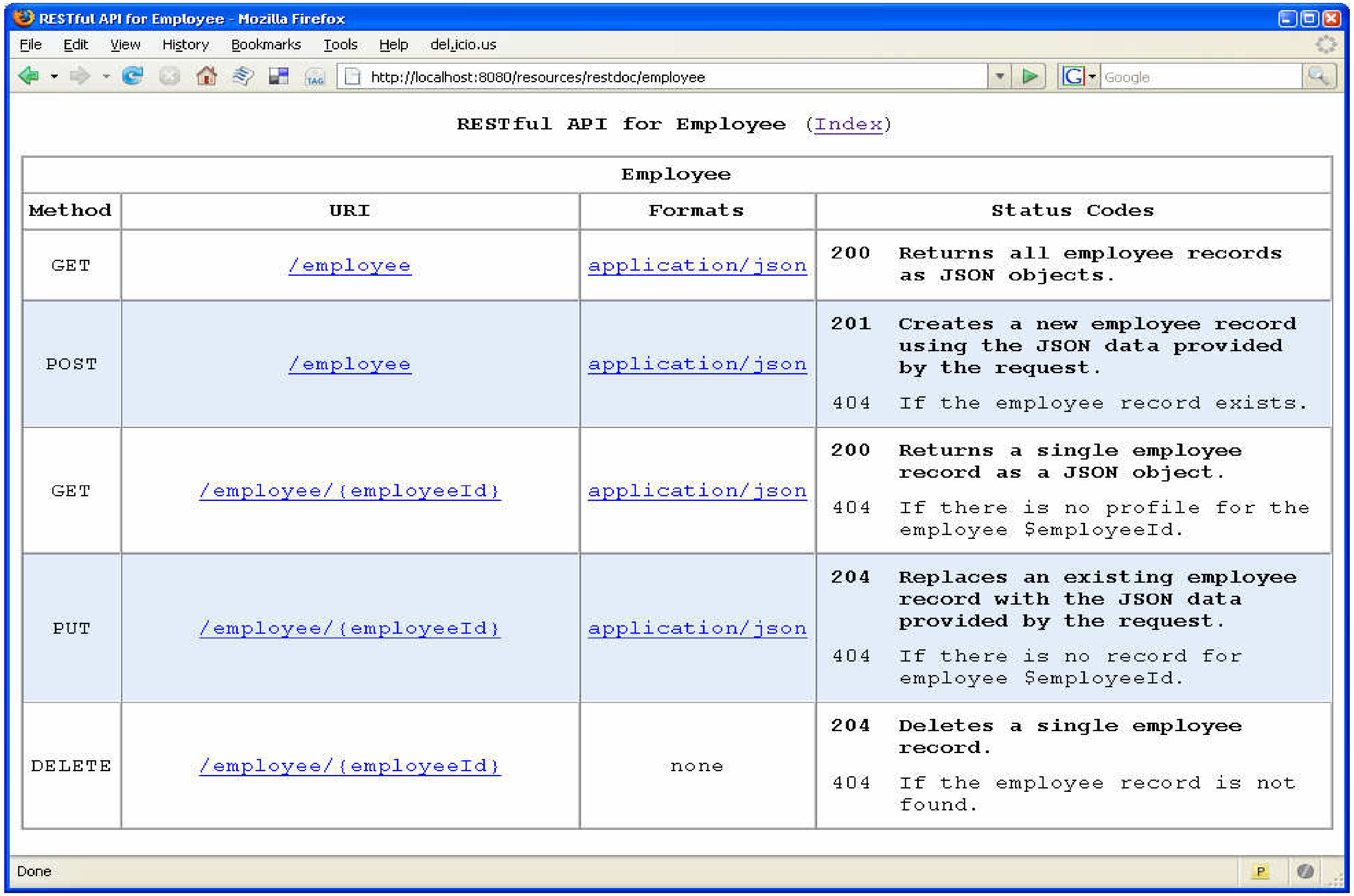
```
request.view = ‘JSON’

request.json.output = object

render()
```

JSON, XML

ATOM

Map	=> Atom entry
List<Map>	=> Atom feed

RESTdoc


RESTful API for Employee - Mozilla Firefox
File Edit View History Bookmarks Tools Help del.icio.us
http://localhost:8080/resources/restdoc/employee
Google
Done

RESTful API for Employee (Index)

Employee				
Method	URI	Formats	Status Codes	
GET	/employee	application/json	200	Returns all employee records as JSON objects.
POST	/employee	application/json	201	Creates a new employee record using the JSON data provided by the request.
			404	If the employee record exists.
GET	/employee/{employeeId}	application/json	200	Returns a single employee record as a JSON object.
			404	If there is no profile for the employee $employeeId.
PUT	/employee/{employeeId}	application/json	204	Replaces an existing employee record with the JSON data provided by the request.
			404	If there is no record for employee $employeeId.
DELETE	/employee/{employeeId}	none	204	Deletes a single employee record.
			404	If the employee record is not found.

An alternative: Zero Resource Model (ZRM)

Model application data

- Constrained set of APIs encourages a **RESTful** application architecture
- Data model that maps well into Atom feeds and JSON formats
- Robust framework for persistence, validation, and serialization

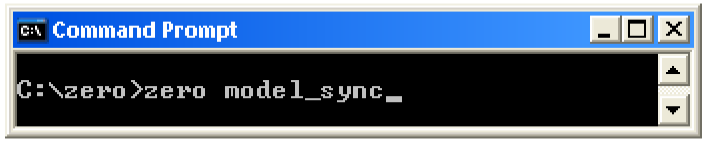
ZRM (continued)

app/models/employee.json

```
{
    "fields" : {
        "first_name": {"type":"string"},
        "last_name": {"type":"string"},
        "location": {"type":"string"}
    }
}
```

app/resources/employee.groovy

```
ZRM.delegate();
```



```
C:\zero>zero model_sync_
```

app/models/fixtures/initial_data.json

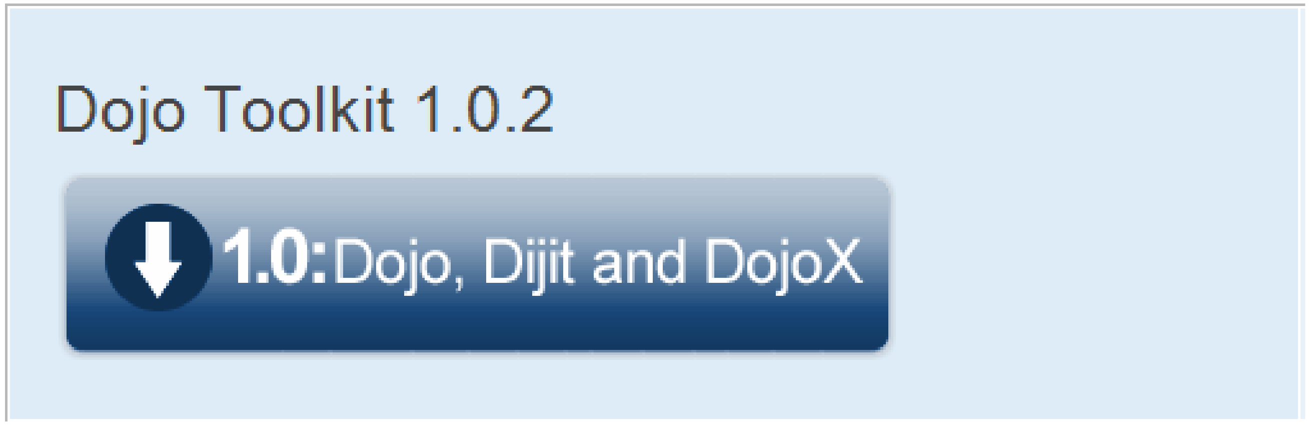
```
[
    {
        "type": "employee",
        "fields": {
                "first_name" :   "Alice",
                "last_name“  :   "Rogers",
                "location"   :   "Seattle"
        }
    },
    {
        "type": "employee",
        "fields": {
                "first_name" :   "Bill",
                "last_name"  :   "Stevens",
                "location"   :   "Seattle"
        }
    },
    {
        "type": "person",
        "fields": {
                "first_name" :   "Cathy",
                "last_name"  :   "Tomlin",
                "location"   :   "Boston"
        }
    }
]
```

WebSphere sMash and RIA

Dojo Broswer Toolkit

Dojo is an Open Source DHTML toolkit written in JavaScript. It builds on several contributed code bases.

- Provides Rich Set of Widgets
- Web UI Framework
- Rich Event handling System
- General Purpose HTML Libraries
- Several other utilities
  - Math, XML to JS parsing, etc…



Dojo Architecture

Base

The kernel of the toolkit wrapped into a **25k** js file (dojo.js). Base bootstraps the toolkit, includes AJAX utilities, class based inheritance, packaging system and more

Core

Provides addition facilities on top of the base for accessing data stores, effects such as wipes/slides, internationalization (i18n) and back-button handling among other things. Separate package keeps base small

Dijit

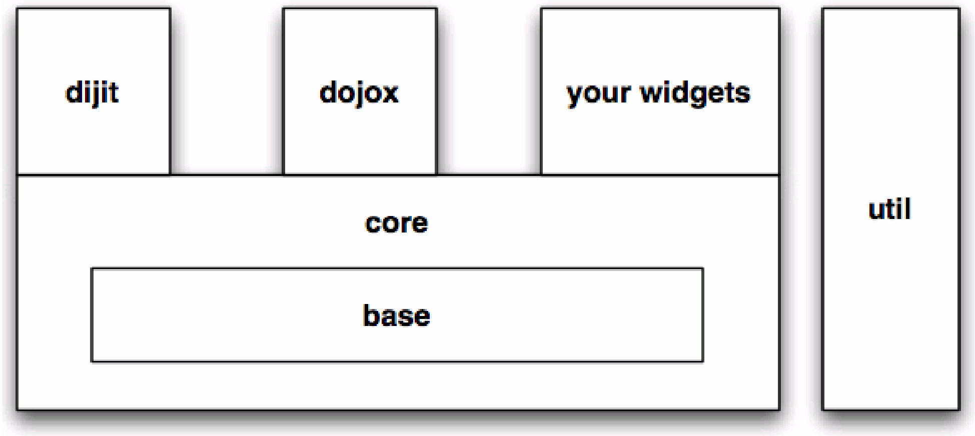
Shorthand for "Dojo widget". Could refer to a single Dojo widget (a dijit) or to the entire component of the toolkit containing all of Dojo's widgets (Dijit)

DojoX

“Dojo Extra" and contains features that stand a chance of one day migrating into Core, Dijit or even a new module. A great proving ground for new features while maintaining standards of core and base.

Util

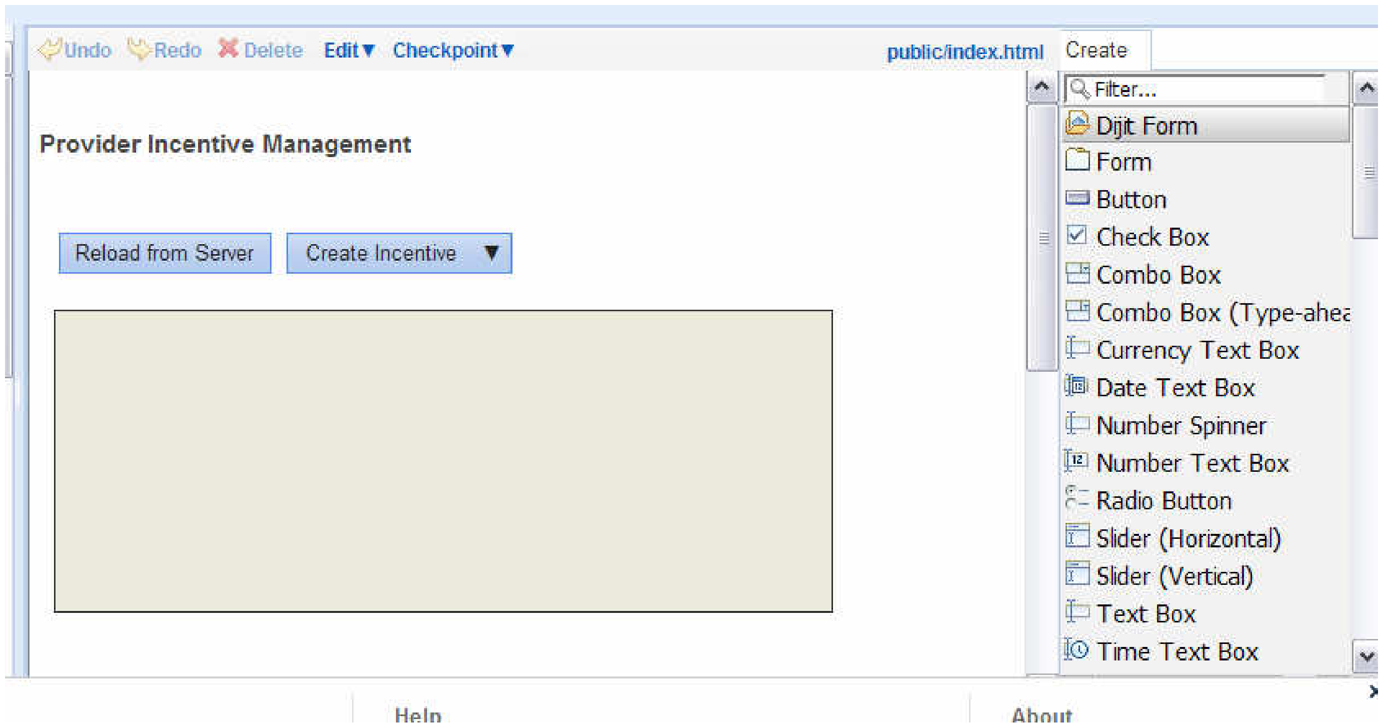
A collection of Dojo utilities (more later)



Web Based IDE Editor for Dojo

Dojo connections with Services through Wires.

Drag and Drop with Dojo Dijits.



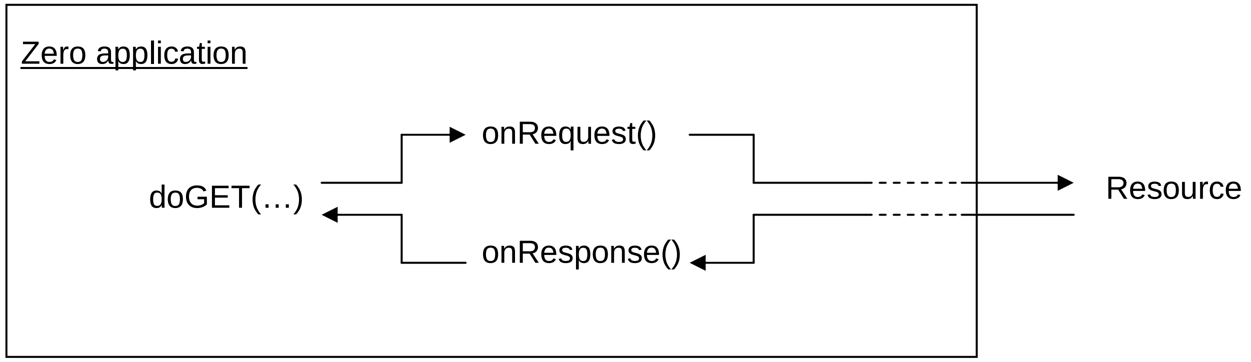
Connections

RESTful API to resources via a variety of protocols

- HTTP/S
- SMTP
- File
- JMS

In-process mediations

- SOAP
- Custom



WebSphere sMash and Flows

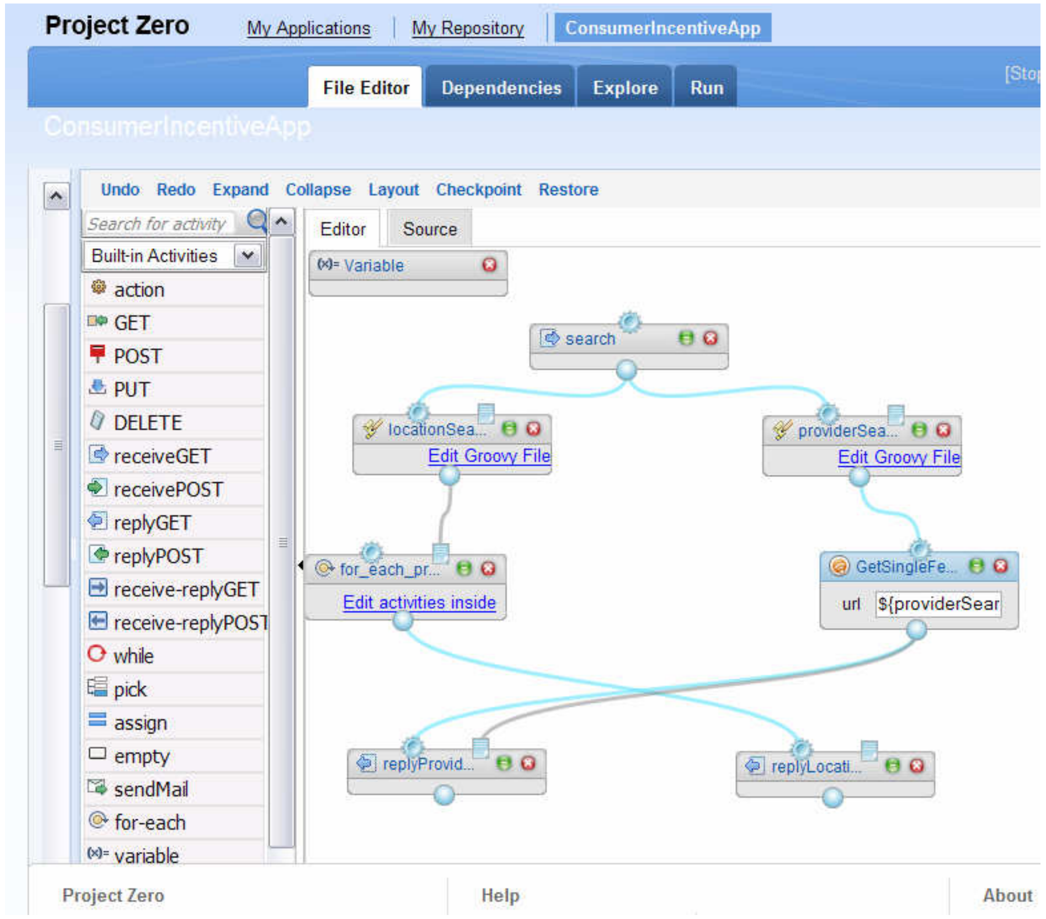
Assemble

Composition of applications by "wiring" REST services using the SPLICE flows. Incorporating both activity and data flows.

A solution may be rapidly assembled by combining existing feeds and services that enrich, sort, and filter data in a pipeline. Either visually or programmatically.

Configure templates to alter pipeline routes, log events along the pipeline, as well as transform data

Adapters to enhance integration with existing systems.



Questions & Answers

AM09

Web 2.0 Development with IBM WebSphere sMash

Matthew Perrins

Executive IT Specialist

Software Group Lab Services

matthew_perrins@uk.ibm.com

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

IBM

Rational. software

What is WebSphere sMash?

WebSphere sMash is an Agile Web Application Platform

- Architected around Dynamic Scripting, REST, Rich Web Interfaces, AJAX, and Feeds
- Optimized for
  - **Speed**
  - **Simplicity**
  - **Agility**

Key Scenarios

- Enables developers to build web 2.0-style applications by easily pulling in, composing, and “cobbling together” pre-existing assets (PHP assets, services, feeds, code snippets) using dynamic scripting languages and simple consumption principles based on REST.
- Leverages existing SOA investments by enabling rapid development of dynamic web applications that are assembled from enterprise assets and publicly available APIs.

WHAT IS PROJECT ZERO?

Project Zero is the development and incubation community for WebSphere sMash

- Live on the Internet since June 2007

Project Zero represents

- The people that build and use WebSphere sMash
- The incubation of new technology that will deliver in future versions of WebSphere sMash
- The community of 3rd party assets that leverage the WebSphere sMash platform

All released versions are called WebSphere sMash

WEBSPHERE SMASH AND THE WEB 2.0 STRATEGY

WebSphere sMash fits perfectly into WebSphere's overall SOA Web 2.0 strategy

The WebSphere Web 2.0 Strategy

- **Unleash enterprise content so it is more easily accessible**
  - WebSphere is REST-enabling its product portfolio, including
    - MQ, Commerce, WSRR, Web 2.0 FP, WPS, WESB, Datapower,
- **Leverage this content by enabling agile web applications**
  - Now that the content is unleashed, you can agilely build web applications to leverage that content as well as content from other backend systems, supplier systems or the web.
  - WebSphere sMash fits here, along with Lotus Mashups and InfoSphere MashupHub, as an agile platform for application creation and deployment
  - WebSphere sMash can also be used here to build and deliver components such as widgets for Lotus Mashups, or feeds for InfoSphere MashupHub
- **Run, manage and host agile applications**
  - Application-centric runtimes (like WebSphere sMash) and management systems (like WebSphere XD) will help you cost effectively run and manage these growing number of these web applications.

PACKAGING

The technology is WebSphere sMash is available in a variety of packages

Package Name	Description
WebSphere sMash	Production version of the WebSphere sMash Platform. Standard IBM commercial license. Available through normal IBM channels.
WebSphere sMash Reliable Transport Extensions	Production version of the extended features in sMash related to messaging and reliable communications. Standard IBM commercial license. Available through normal IBM channels.
WebSphere sMash Developer Edition (DE)	This is the community version of the exact same code you get with WebSphere sMash. WebSphere sMash DE represents the shipped and stable version of the product for developers to use to build applications.
Project Zero	This is the community version of the latest and greatest unreleased technology that is not in a WebSphere sMash version yet. This is the bleeding edge incubation of new features.

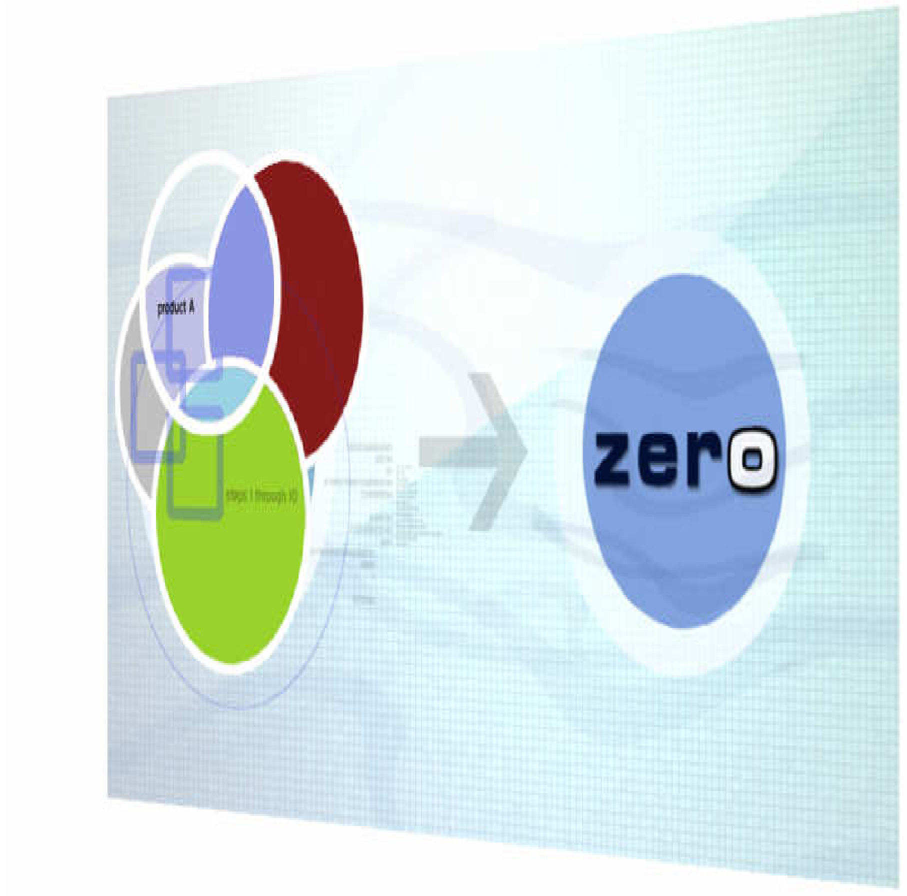
Technology Overview

THE WEBSPHERE SMASH CORE VALUES

Speed

Simplicity

Agility



CORE APPLICATION CONSTRUCTS

EASY FOR DEVELOPERS TO ACCESS, LEARN, AND USE

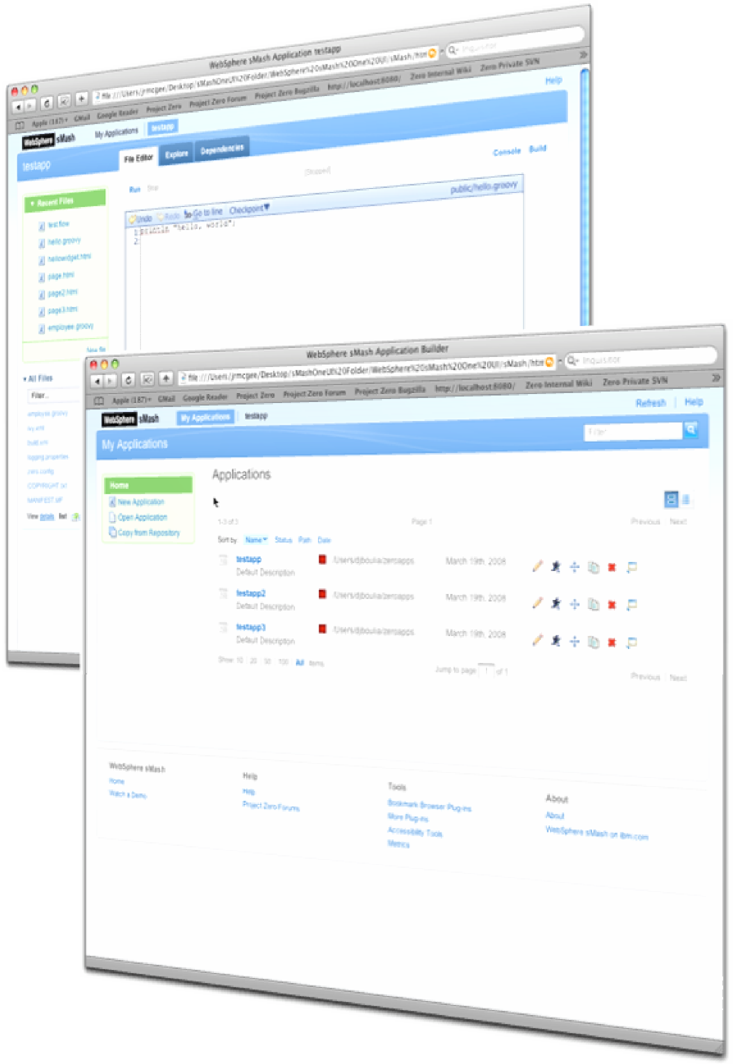
Dynamic Scripting and Templates

Effortless creation of RESTful Services and Data Feeds (RSS, ATOM)

Simple Event-based execution environment

State externalized into a shared memory space (Global Context)

Repository of pre-built Services and Libraries provides useful building blocks



DYNAMIC SCRIPTING

- WebSphere sMash is a dynamic scripting platform
- Application Logic is created in one of two scripting languages
  - Groovy (for people that prefer Java)
  - PHP (for the 3 Million existing PHP programmers)
- Java is positioned as the "system" language
  - Mostly used to implement system extensions and application libraries
  - Entire applications can be written in Java, if desired
    - Requires more configuration

APPLICATION CENTRIC RUNTIME

WebSphere sMash is an application-centric runtime

- You create an application and run it
- You do not package an application and deploy it to a multi-application server
- Each application runs in its own process (JVM)
- Runtime is designed to be short lived
  - Update recycles after idle timeout or max number of requests

WebSphere sMash is a full stack runtime

- Everything needed to run the application is built in
  - Including the HTTP stack
- No external proxy or web server is required
- Does not deploy as a WAR file inside another JEE container
- An external proxy is used for clustering and multi-app routing

PHP SUPPORT

- Gartner predict that within 5 years 60% of the 5.5 million PHP programmers will work in corporate IT. (Up from 13% of 3M today).
- The PHP runtime is built on top of a standard JVM
  - Supports use of many PHP Extensions
    - XAPI-C interface allows C-based extensions
    - XAPI-J interface allows Java based extensions
  - Supports bridging between Java and PHP
  - Currently supports a subset of PHP
    - The goal is maximum re-use of existing PHP scripts
- PHP runtime provided directly by WebSphere sMash, not php.net
- The goal of PHP support is about unleashing the 3 Million PHP programmers together with the vast library of existing PHP script code and extensions and bringing it to the sMash programming model
- A number of popular PHP application now run on the sMash PHP runtime, including
  - **phpBB**
  - **SugarCRM**

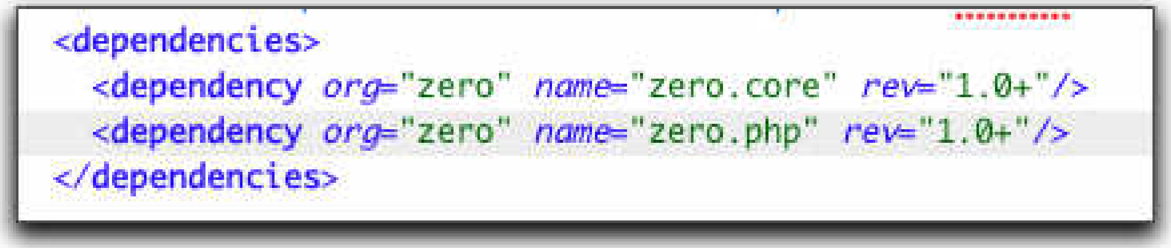
MODULAR ARCHITECTURE

WebSphere sMash applications are based on a very small core

- 5.4 MBytes (includes Groovy).
  - PHP adds additional 14.5 Mbytes
    - Contains 3 platforms currently
- Core provides all of the framework and runtime support, including HTTP transport

Additional features provided in downloadable modules

- Applications declare a dependency on desired features (using Ivy)
- A package management system manages your dependencies, including:
  - The ability to share dependencies on a machine
  - The ability to demand load missing dependencies from the network
  - The ability to manage updates to dependencies that you are using



```
<dependencies>
  <dependency org="zero" name="zero.core" rev="1.0+"/>
  <dependency org="zero" name="zero.php" rev="1.0+"/>
</dependencies>
```

SIMPLE DEPLOYMENT

Essentially the deployment is Zip and Copy

No machine specific information bound into the application

Default mode is shared dependencies

- Application dependencies are load for the deployment machines local repository and pulled off the network if needed

Standalone mode is supported as well

- All application dependencies are included in the ZIP and nothing is needed on the target machine except a JVM

Provides a packaging command to simplify the creation of the ZIP file for deployment

- zero package for shared mode
- zero package standalone for standalone mode

RUNTIME CHARACTERISTICS

- **Instant On**
  - Application Available for Service in less than 1 sec
    - 0.672 seconds on a MacBook Pro
  - Application JVM starts in about 1 second
    - 1.3 seconds on a MacBook Pro
- **Clean**
  - Graceful recovery, isolation, tolerates “bad” code
  - Short lived processes
    - Runs for fixed number of requests or idle timeout then restarts
    - No state lost on restart
- **Cheap**
  - Cost effective to run in small and large quantities
  - Idle Application Footprint ~380 Kbytes
  - Running Application JVM ~28 Mbytes TODAY
- **Supported on “stock” JVM**
  - IBM, Sun, Mac, etc - Any JSE 5 or 6 JVM

BUILT-IN DEVELOPMENT TOOLING

WEBSPERE SMASH LETS DEVELOPERS BUILD APPLICATION DIRECTLY ON THE WEB

Browser-Based Development IDE

Built as a sMash application

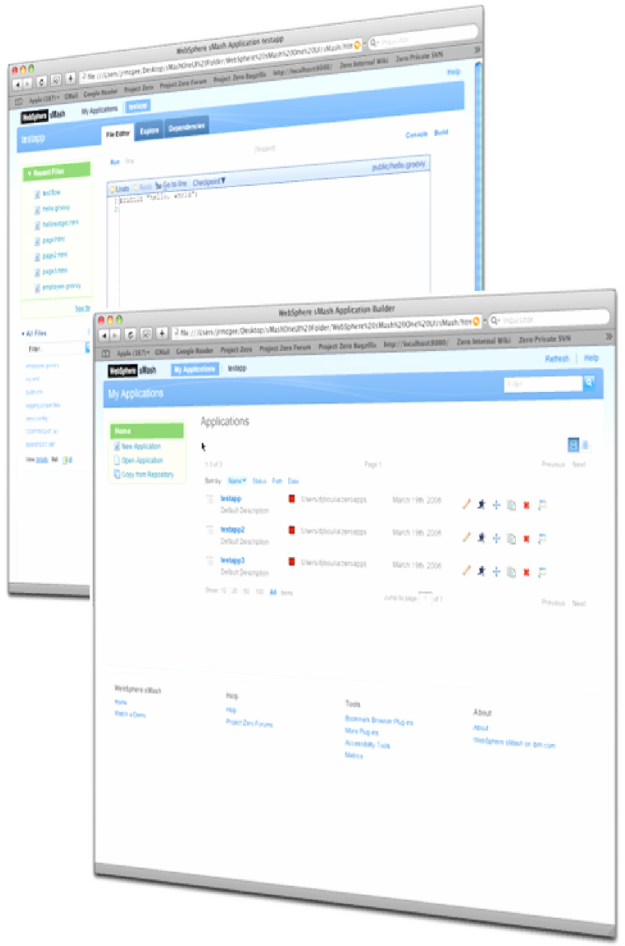
Provides full development lifecycle for Zero applications

- Create, Edit, Test

Provides Visual Editors for Activities and Web Page construction

- Including a DOJO-enabled page editor

Basic Eclipse-based tooling also available if required



WEBSPHERE SMASH ACTIVITIES
LETS DEVELOPERS VISUALLY "MASH-UP" SERVICES AND FEEDS

Assemble-style Development

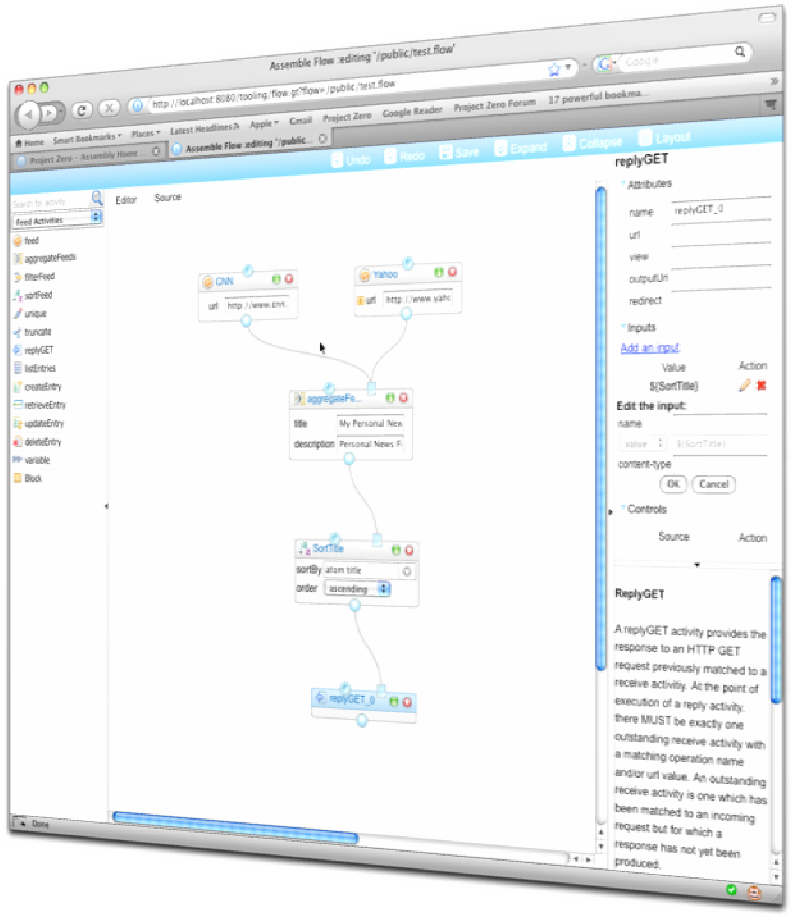
Compose applications by "wiring" together REST services

Visually or programmatically combine existing feeds and services that enrich, sort, and filter data in a pipeline

Configure templates to alter pipeline routes, log events along the pipeline

Numerous built-in activities, including

Get Feed, Call Service, Aggregate, Sort, Transform, Filter, Send Mail, XSLT, Conditionals, Loops



ReplyGET

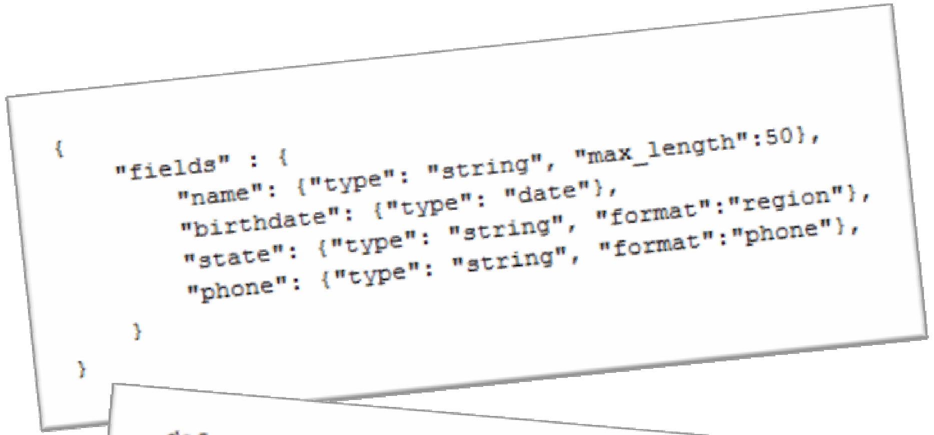
A replyGET activity provides the response to an HTTP GET request previously matched to a receive activity. At the point of execution of a reply activity, there MUST be exactly one outstanding receive activity with a matching operation name and/or url value. An outstanding receive activity is one which has been matched to an incoming request but for which a response has not yet been produced.

RAPIDLY EXPOSE DATA RESTFULLY

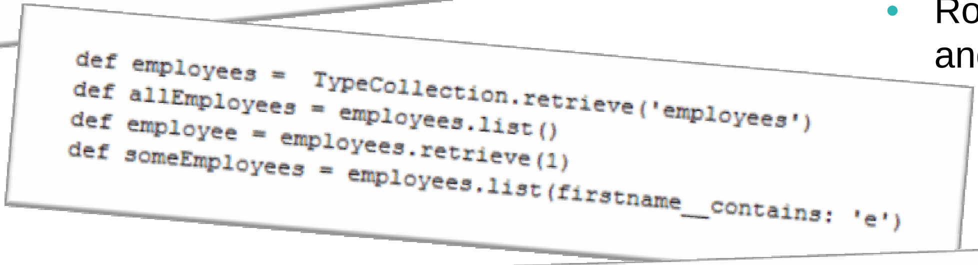
ZERO RESOURCE MODEL ENABLES DEVELOPERS WITH A SIMPLE PROGRAMMATIC AND HTTP DATA API

Model application data

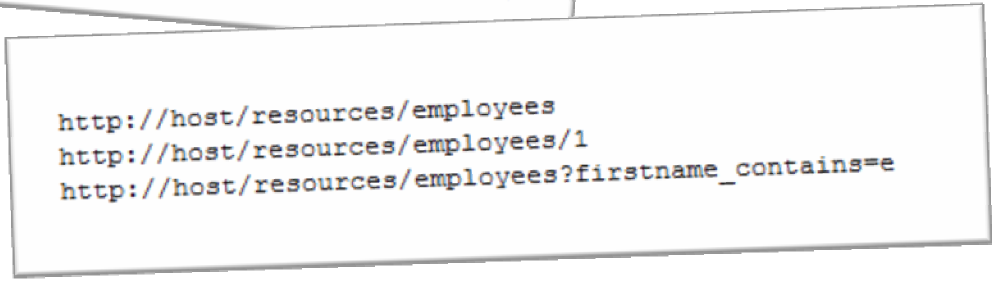
- Constrained set of APIs encourage a RESTful application architecture
- Data model that maps well into Atom feeds and JSON formats
- Robust framework for persistence, validation, and serialization



```
{
    "fields" : {
        "name": {"type": "string", "max_length":50},
        "birthdate": {"type": "date"},
        "state": {"type": "string", "format":"region"},
        "phone": {"type": "string", "format":"phone"},
    }
}
```



```
def employees =  TypeCollection.retrieve('employees')
def allEmployees = employees.list()
def employee = employees.retrieve(1)
def someEmployees = employees.list(firstname__contains: 'e')
```



```
http://host/resources/employees
http://host/resources/employees/1
http://host/resources/employees?firstname_contains=e
```

AVAILABLE MODULES

- There are approximately 65 modules available currently
- Modules provide function in many categories
  - Data Formats (JSON, ATOM, RSS, XML)
  - Data Access
  - Resource Modeling
  - Security / Content Filtering
  - Activity Flows
  - Services
    - Amazon ECS, Flickr, Weather, etc
  - Utilities (such as HTML parsing)
  - Management Tools
  - Development Tooling
  - Reliable Transport Engine for Messaging Interactions

COMMUNITY DRIVEN COMMERCIAL DEVELOPMENT

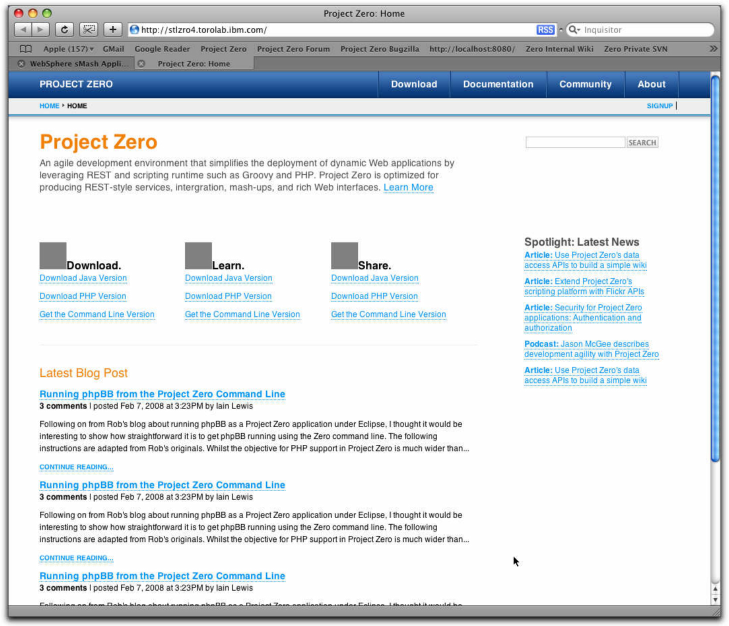
Evolve the core platform based on developer feedback

Commercial development using a transparent development process

Enabled via an external web site providing:

- *A focal point for all sMash development activities*
  - Expose the IBM development process to the external developer community
  - All design decisions are discussed and communicated via external forums
  - Registered users can post comments and feedback to the forums
- *Frictionless download of latest code and documentation*
  - Registration not required for binary downloads
  - Latest builds immediately available to developers
  - Source code can be viewed by registered users

http://www.projectzero.org



Project Zero

An agile development environment that simplifies the deployment of dynamic Web applications by leveraging REST and scripting runtime such as Groovy and PHP. Project Zero is optimized for producing REST-style services, intergration, mash-ups, and rich Web interfaces. Learn More

Download.
Download Java Version
Download PHP Version
Get the Command Line Version

Learn.
Download Java Version
Download PHP Version
Get the Command Line Version

Share.
Download Java Version
Download PHP Version
Get the Command Line Version

Spotlight: Latest News

Article: Use Project Zero's data access APIs to build a simple wiki

Article: Extend Project Zero's scripting platform with Flickr APIs

Article: Security for Project Zero applications: Authentication and authorization

Podcast: Jason McGee describes development agility with Project Zero

Article: Use Project Zero's data access APIs to build a simple wiki

Latest Blog Post

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

Following on from Rob's blog about running phpBB as a Project Zero application under Eclipse, I thought it would be interesting to show how straightforward it is to get phpBB running using the Zero command line. The following instructions are adapted from Rob's originals. Whilst the objective for PHP support in Project Zero is much wider than...

CONTINUE READING...

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

Following on from Rob's blog about running phpBB as a Project Zero application under Eclipse, I thought it would be interesting to show how straightforward it is to get phpBB running using the Zero command line. The following instructions are adapted from Rob's originals. Whilst the objective for PHP support in Project Zero is much wider than...

CONTINUE READING...

Running phpBB from the Project Zero Command Line
3 comments | posted Feb 7, 2008 at 3:23PM by Iain Lewis

PROJECTZERO.ORG
LOWERING THE BARRIERS TO COMMUNITY ACCESS

Anonymous Visitors can…

Browse the site
View Wiki content
Read Forums
Search the Bug Database
Read Blogs
Download Binary Drivers*

Focused on easy access

- Internet web site
- Free access to the platform

Registered Users can…

Post to the Forum
Submit Bug Reports
Submit Feature Requests
Comment on Blog Posts
Access Source Code*

Focused on feedback

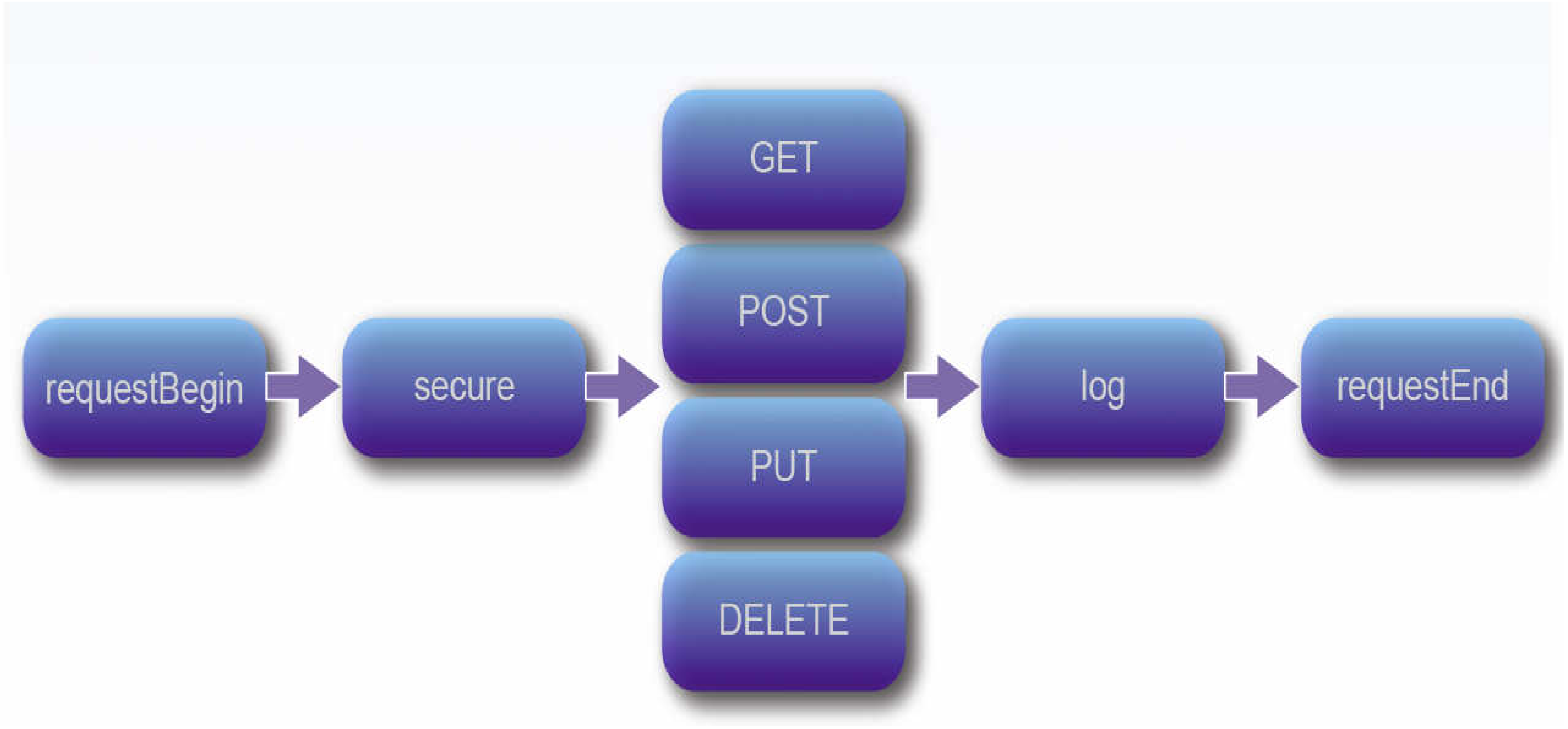
- Simple, free registration process

* Requires acceptance of an IBM license agreement

Core Programming Model

Events

- All behavior in the system is modeled as a set of event
  - Applications are built by handling these events and providing desired behavior
  - Similar to AJAX model or classic UI programming



Event Handlers

All handlers are stateless

Can be implemented in Groovy, PHP, and Java

Groovy

```
println “Hello World”
```

```
def onGET()
{
   println “Hello World”
}
```

PHP

```
function onGET()
{
   echo “Hello World”;
}
```

Java

```
public void onGET()
{
   PrintWriter writer = (PrintWriter)zget(“/request/writer”);
   writer.println(“Hello World”);
}
```

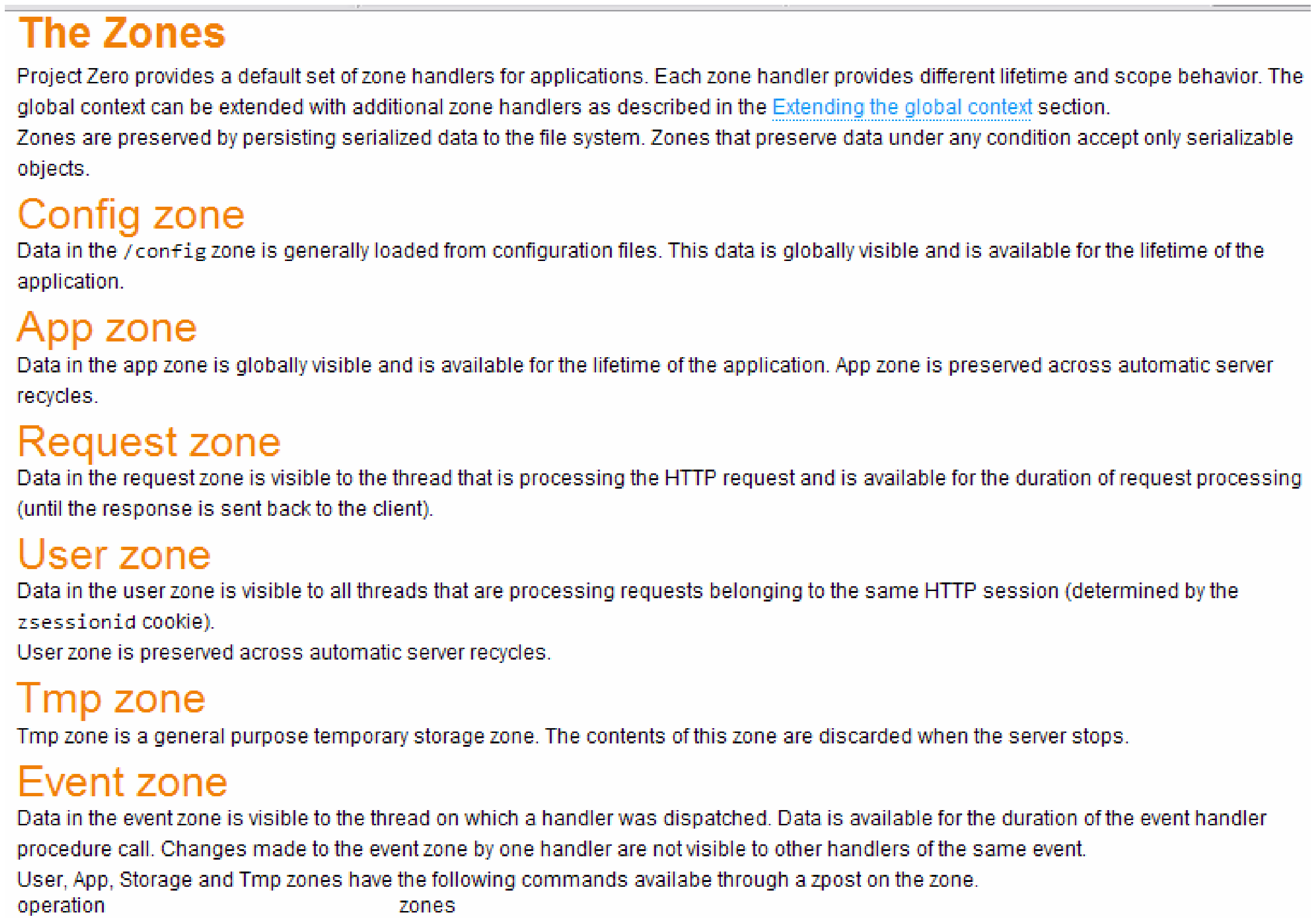
```
/config/handlers += {
   “events” : “GET”,
   “handler” : “HelloWorld.class”,
   “conditions” : [“/request/path matches /hello”]
}
```

Global Context – State Management

- **The Global Context (GC) provides access to and management of all application state**
  - Conceptually a map of data
- **Externalizes all state from the application logic**
  - Enables the restart ability of the JVM without data loss
  - Enables clustering and scaling to be added transparently
- Simplifies and unifies access to application state and data structures and simplifies state passing within the application
- Contains information provided by both the runtime (such as request parameters) and by the application

Global Context Zone

Divided into zones representing different data lifecycles



The Zones

Project Zero provides a default set of zone handlers for applications. Each zone handler provides different lifetime and scope behavior. The global context can be extended with additional zone handlers as described in the Extending the global context section.

Zones are preserved by persisting serialized data to the file system. Zones that preserve data under any condition accept only serializable objects.

Config zone

Data in the `/config` zone is generally loaded from configuration files. This data is globally visible and is available for the lifetime of the application.

App zone

Data in the app zone is globally visible and is available for the lifetime of the application. App zone is preserved across automatic server recycles.

Request zone

Data in the request zone is visible to the thread that is processing the HTTP request and is available for the duration of request processing (until the response is sent back to the client).

User zone

Data in the user zone is visible to all threads that are processing requests belonging to the same HTTP session (determined by the `zsessionid` cookie).

User zone is preserved across automatic server recycles.

Tmp zone

Tmp zone is a general purpose temporary storage zone. The contents of this zone are discarded when the server stops.

Event zone

Data in the event zone is visible to the thread on which a handler was dispatched. Data is available for the duration of the event handler procedure call. Changes made to the event zone by one handler are not visible to other handlers of the same event.

User, App, Storage and Tmp zones have the following commands availabe through a zpost on the zone.

operation	zones

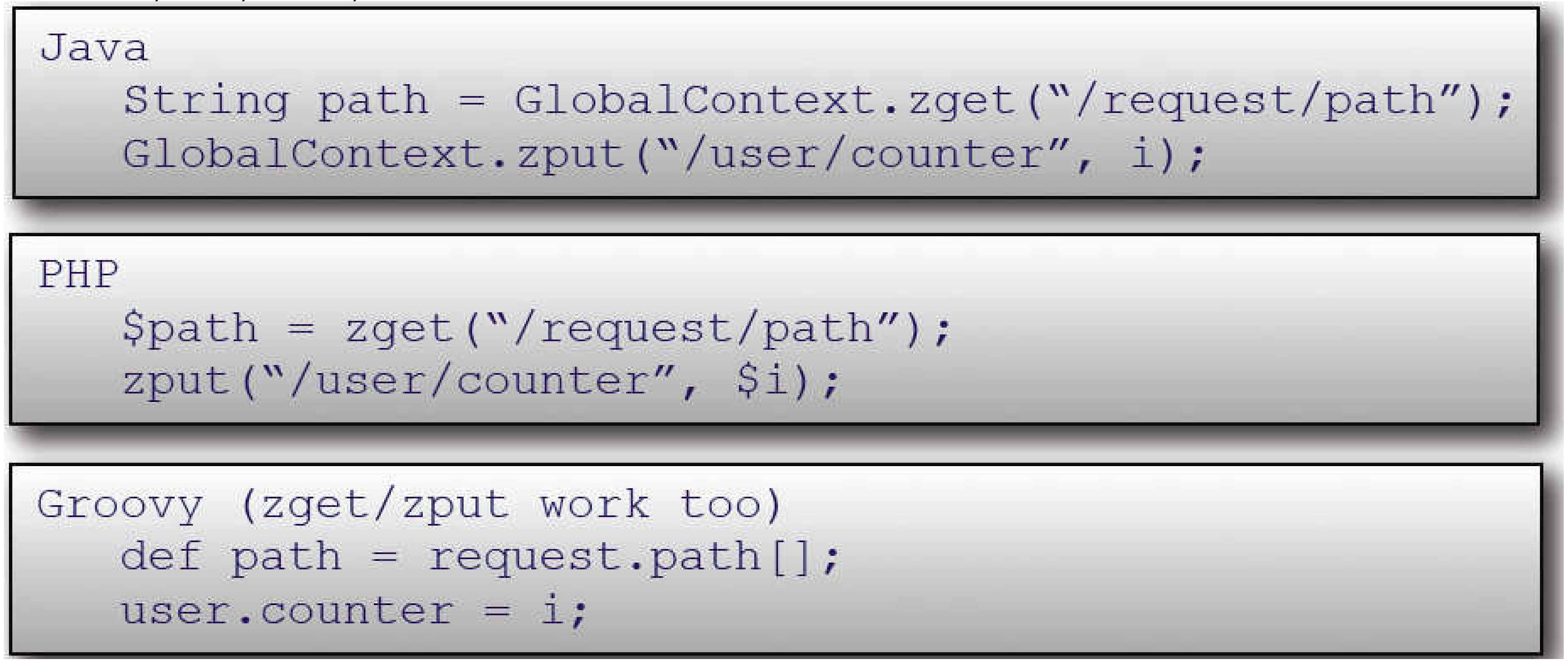
Accessing the Global Context

Data is organized by a URI structure

- First part of URI is always the Zone name
  - /app, /user, /request, /config, /event, /client

Access is modeled after REST

- GET, PUT, POST, DELETE



```
Java
    String path = GlobalContext.zget("/request/path");
    GlobalContext.zput("/user/counter", i);
```

```
PHP
    $path = zget("/request/path");
    zput("/user/counter", $i);
```

```
Groovy (zget/zput work too)
    def path = request.path[];
    user.counter = i;
```

Value Pathing

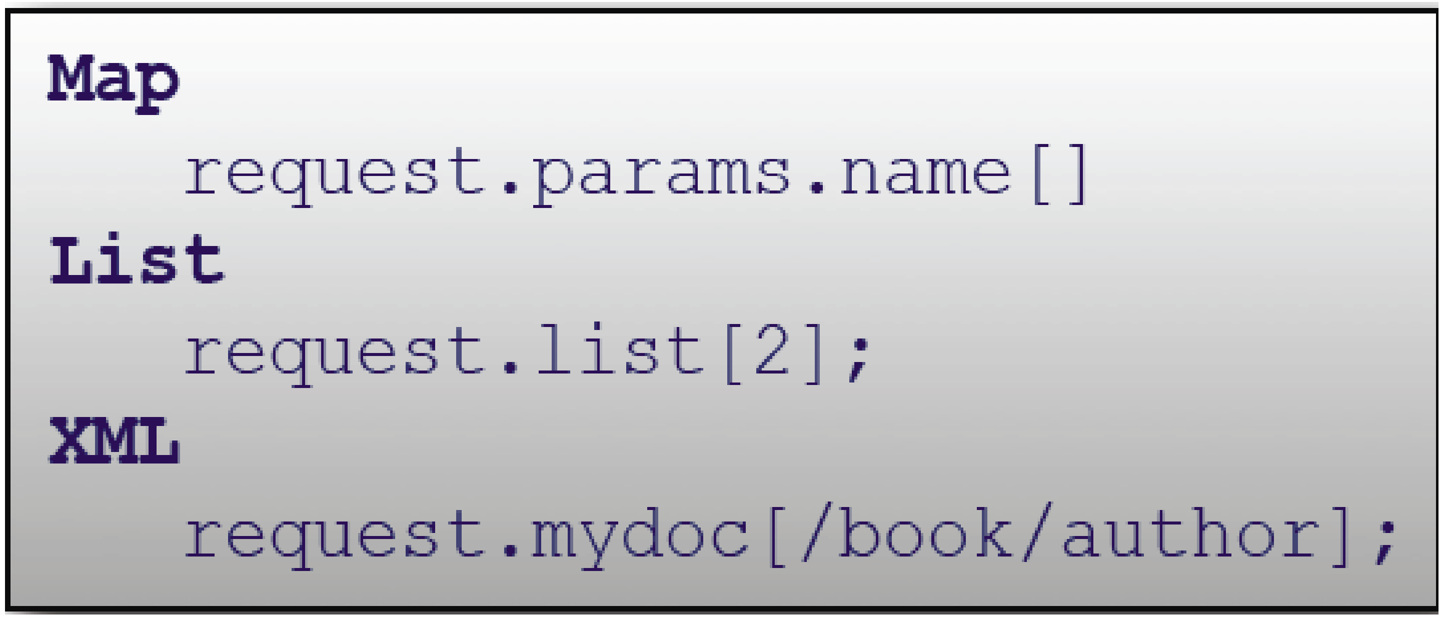
The GC provides simplified access to certain data structures

Called ***Value Pathing***

Understands

Maps, List, Objects, XML, JSON

□Allows read and write access to internals of the structure through the GC address



Map

```
    request.params.name[]
```

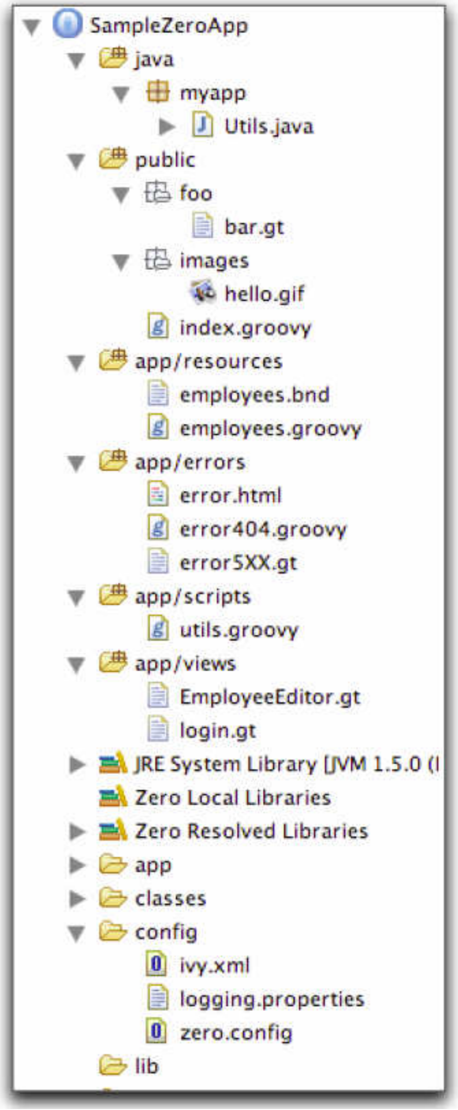
List

```
    request.list[2];
```

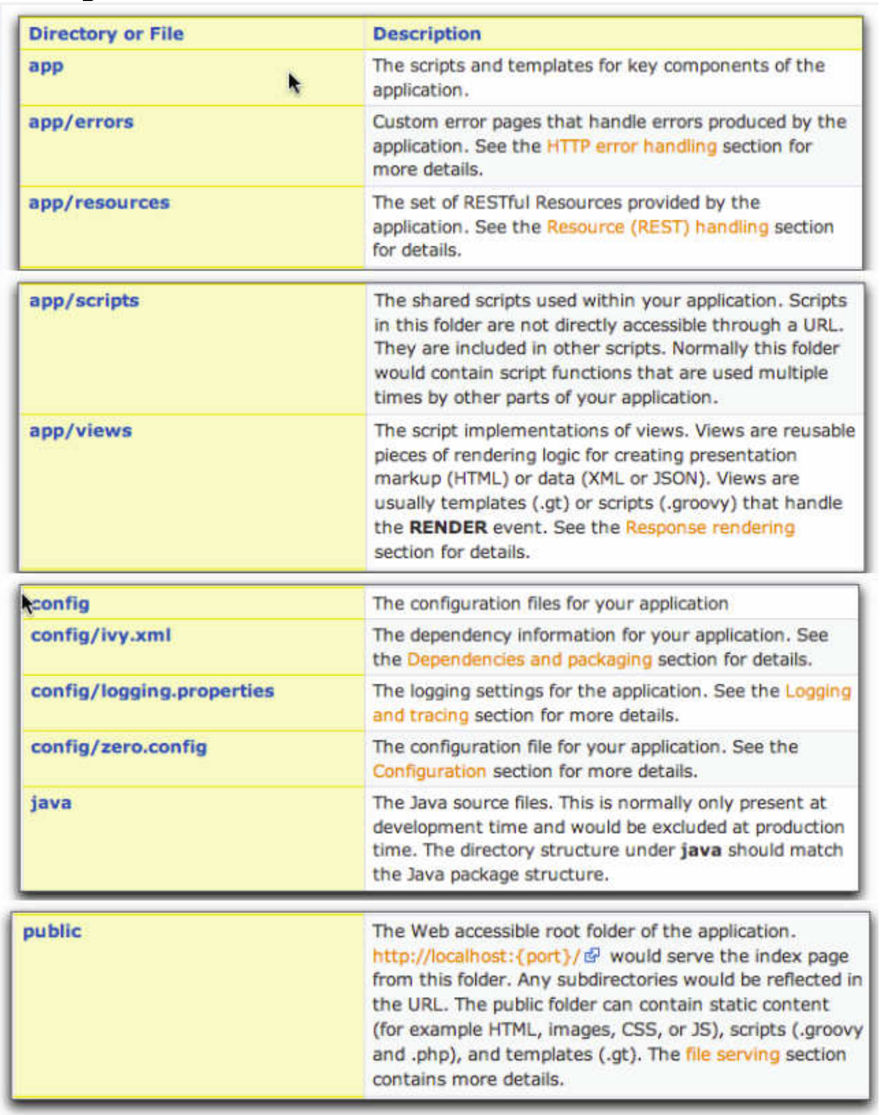
XML

```
    request.mydoc[/book/author];
```

Application Directory Layout



```
▼ SampleZeroApp
  ▼ java
    ▼ myapp
      ▶ Utils.java
  ▼ public
    ▼ foo
        bar.gt
    ▼ images
        hello.gif
      index.groovy
  ▼ app/resources
      employees.bnd
      employees.groovy
  ▼ app/errors
      error.html
      error404.groovy
      error5XX.gt
  ▼ app/scripts
      utils.groovy
  ▼ app/views
      EmployeeEditor.gt
      login.gt
  ▶ JRE System Library [JVM 1.5.0 (I
    Zero Local Libraries
  ▶ Zero Resolved Libraries
  ▶ app
  ▶ classes
  ▼ config
      ivy.xml
      logging.properties
      zero.config
    lib
```

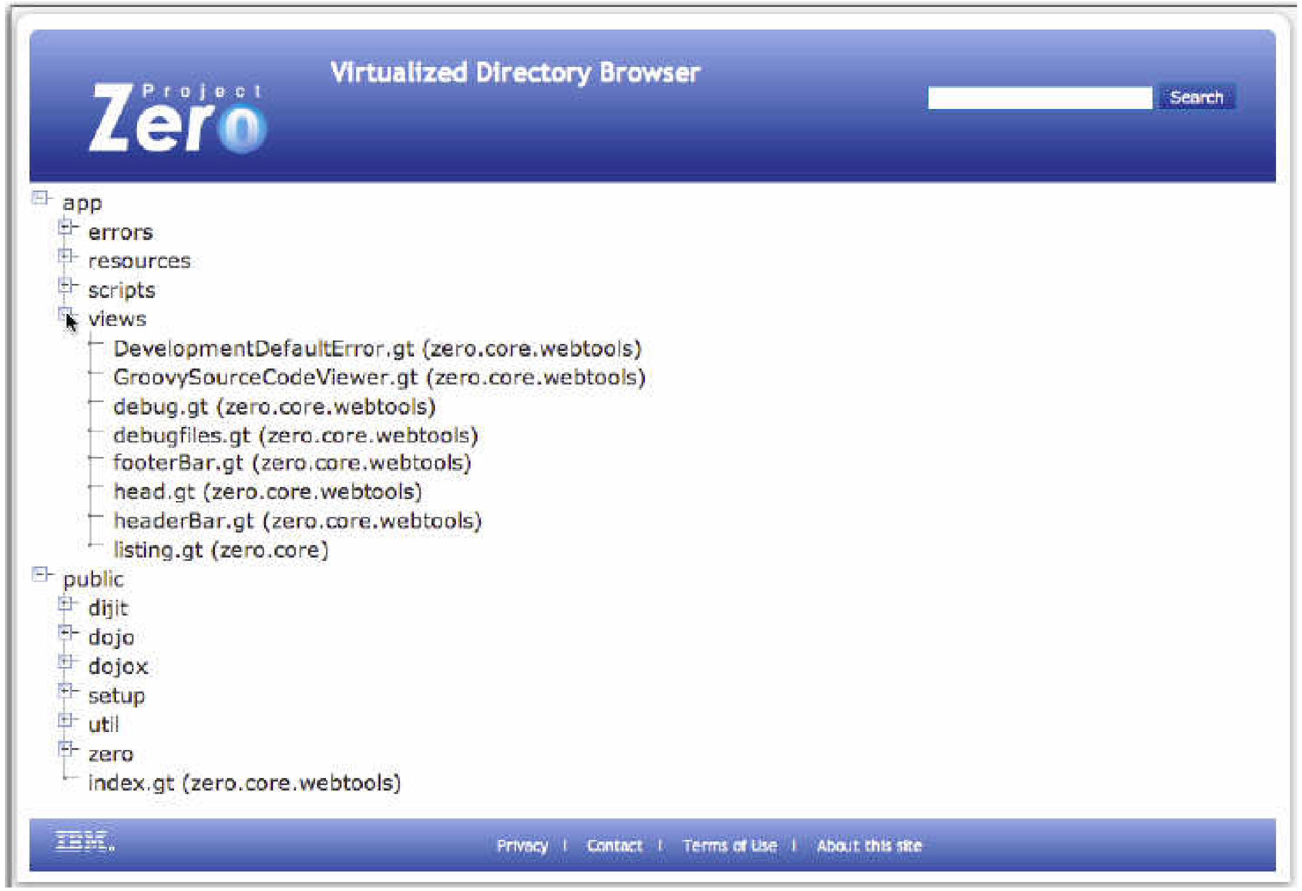


Directory or File	Description
app	The scripts and templates for key components of the application.
app/errors	Custom error pages that handle errors produced by the application. See the HTTP error handling section for more details.
app/resources	The set of RESTful Resources provided by the application. See the Resource (REST) handling section for details.
app/scripts	The shared scripts used within your application. Scripts in this folder are not directly accessible through a URL. They are included in other scripts. Normally this folder would contain script functions that are used multiple times by other parts of your application.
app/views	The script implementations of views. Views are reusable pieces of rendering logic for creating presentation markup (HTML) or data (XML or JSON). Views are usually templates (.gt) or scripts (.groovy) that handle the **RENDER** event. See the Response rendering section for details.
config	The configuration files for your application
config/ivy.xml	The dependency information for your application. See the Dependencies and packaging section for details.
config/logging.properties	The logging settings for the application. See the Logging and tracing section for more details.
config/zero.config	The configuration file for your application. See the Configuration section for more details.
java	The Java source files. This is normally only present at development time and would be excluded at production time. The directory structure under **java** should match the Java package structure.
public	The Web accessible root folder of the application. http://localhost:{port}/ would serve the index page from this folder. Any subdirectories would be reflected in the URL. The public folder can contain static content (for example HTML, images, CSS, or JS), scripts (.groovy and .php), and templates (.gt). The file serving section contains more details.

Virtualized Directory

Project Zero provides seamless integration of directories across an application and its dependencies, while maintaining each as separate entities.

All artifacts are searched within both the application and its declared dependencies


Project Zero
Virtualized Directory Browser
Search
IBM.
Privacy | Contact | Terms of Use | About this site

```
app
  errors
  resources
  scripts
  views
    DevelopmentDefaultError.gt (zero.core.webtools)
    GroovySourceCodeViewer.gt (zero.core.webtools)
    debug.gt (zero.core.webtools)
    debugfiles.gt (zero.core.webtools)
    footerBar.gt (zero.core.webtools)
    head.gt (zero.core.webtools)
    headerBar.gt (zero.core.webtools)
    listing.gt (zero.core)
public
  dijit
  dojo
  dojox
  setup
  util
  zero
  index.gt (zero.core.webtools)
```

Configuration

Zero configuration file: zero.config

The config/zero.config file is processed at the start of a Zero application.

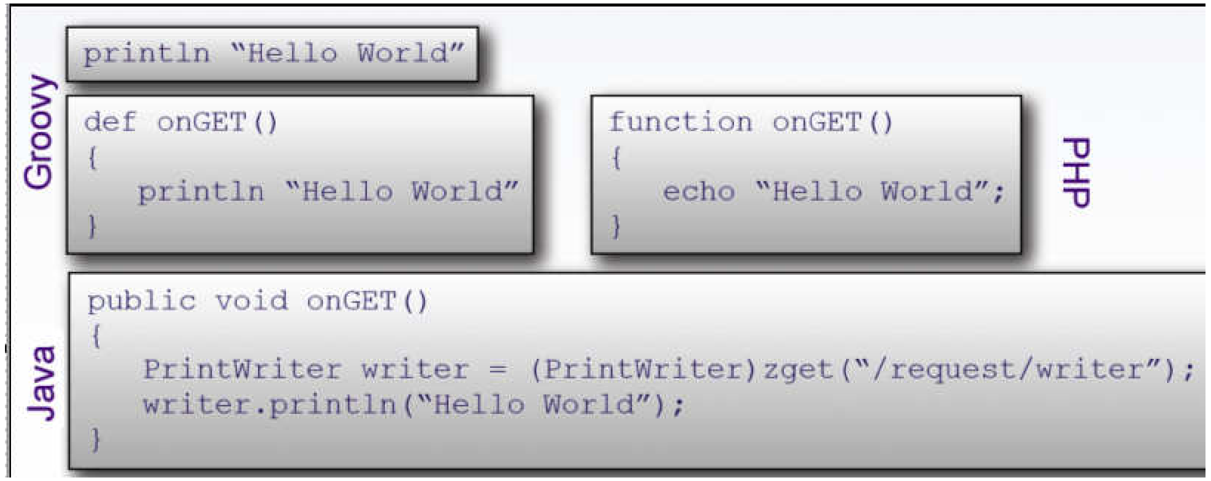
The content of a config/zero.config file is organized into "stanzas" of related key/value pairs. Stanzas are associated with directives, such as "store to the Global Context" and "include another configuration file."

```
# Value set
/config/http/port = 8080

# List set
/config/resources/defaultExtensions = [".groovy"]

# List append
/config/bindings/.groovy +=
["zero.core.groovysupport.bindings.DefaultBindings"]

# Map set
/config/test/map = { "a" : "b", "c" : "d" }

# Map append
/config/test/mapappend += { "a" : "b", "c" : "d" }
/config/test/mapappend += { "x" : "y", "w" : "z" }

# Event handler
/config/handlers += {
  "events" : "GET",
  "handler" : "custom.Handler.class"
}

# Value reference (insert value read at config-load time)
/config/property/myPrefix = "/foo/bar"
/config/test/value = "${/config/property/myPrefix}/bat"

# Variable set/value reference
myPrefix = "/foo/bar"
/config/test/value = "${myPrefix}/bat"

# Include
@include "${/config/dependencies/zero.core}/config/security/
form.config"
{ "formLoginPage" : "/login" }
```

Rendering

Direct

Indirect

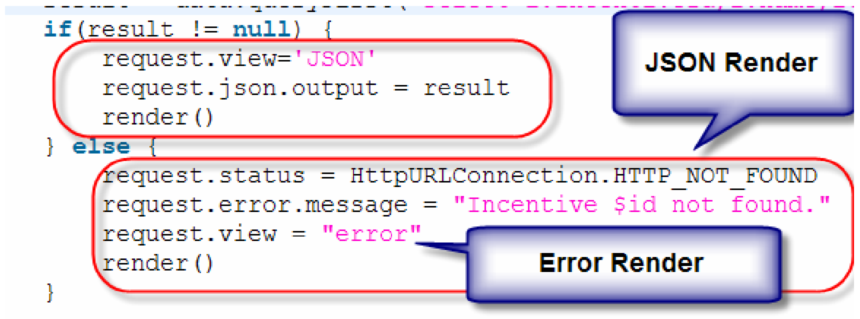


```
println "Hello World"
```

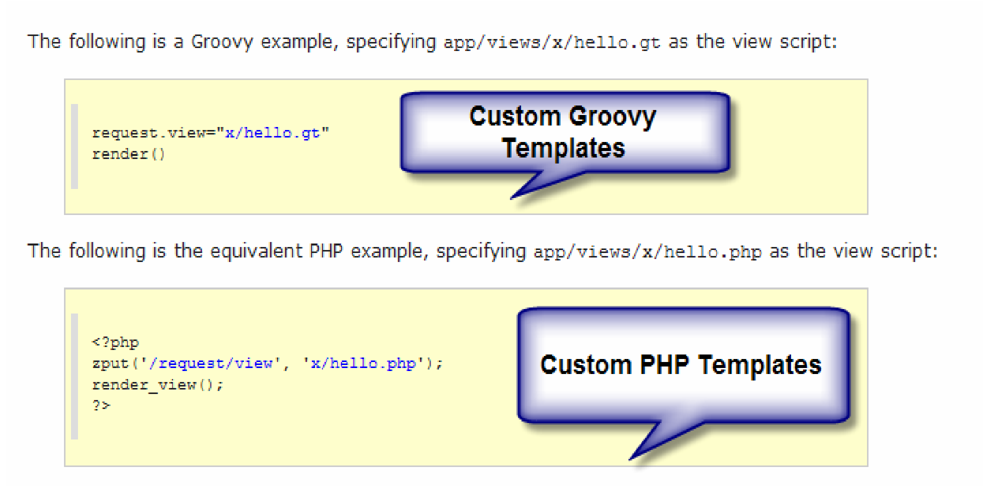
```
def onGET()
{
   println "Hello World"
}
```

```
function onGET()
{
   echo "Hello World";
}
```

```
public void onGET()
{
   PrintWriter writer = (PrintWriter)zget("/request/writer");
   writer.println("Hello World");
}
```



```
if(result != null) {
    request.view='JSON'
    request.json.output = result
    render()
} else {
    request.status = HttpURLConnection.HTTP_NOT_FOUND
    request.error.message = "Incentive $id not found."
    request.view = "error"
    render()
}
```



The following is a Groovy example, specifying `app/views/x/hello.gt` as the view script:

```
request.view="x/hello.gt"
render()
```

The following is the equivalent PHP example, specifying `app/views/x/hello.php` as the view script:

```
<?php
zput('/request/view', 'x/hello.php');
render_view();
?>
```

REST

What is REST?

Representational State Transfer

Architectural model on which the World Wide Web is based

Principles of REST

- Resource-centric approach
- All relevant resources are addressable via URIs
- Uniform access via HTTP – GET, POST, PUT, DELETE
- Content-type negotiation allows retrieving alternative representations from same URI

REST-style services

- are easy to access from code running in web browsers, any other client or servers
- can serve multiple representations of the same resource

Accessibility for Developers

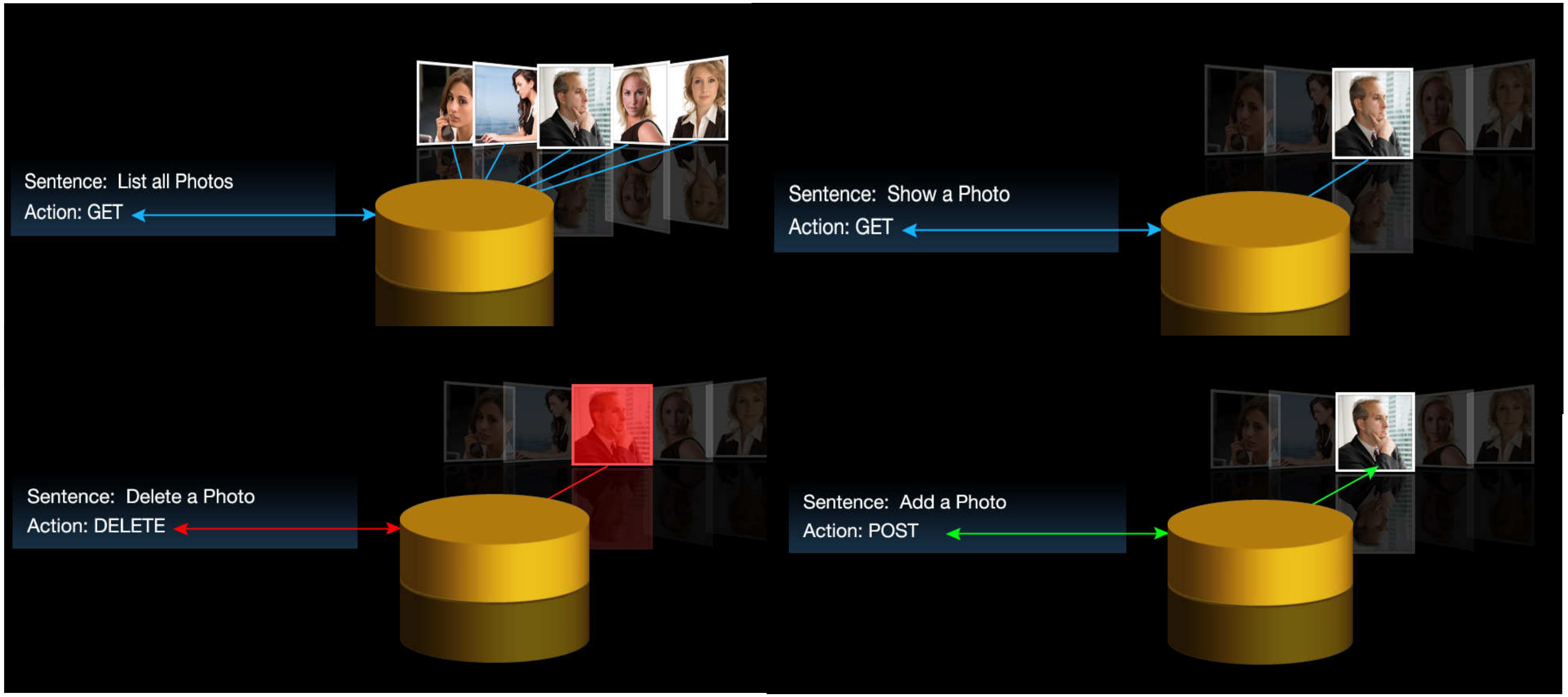
Simply exposing services from the enterprise as URLs and Feeds

A RESTful Web service is formed like a sentence:

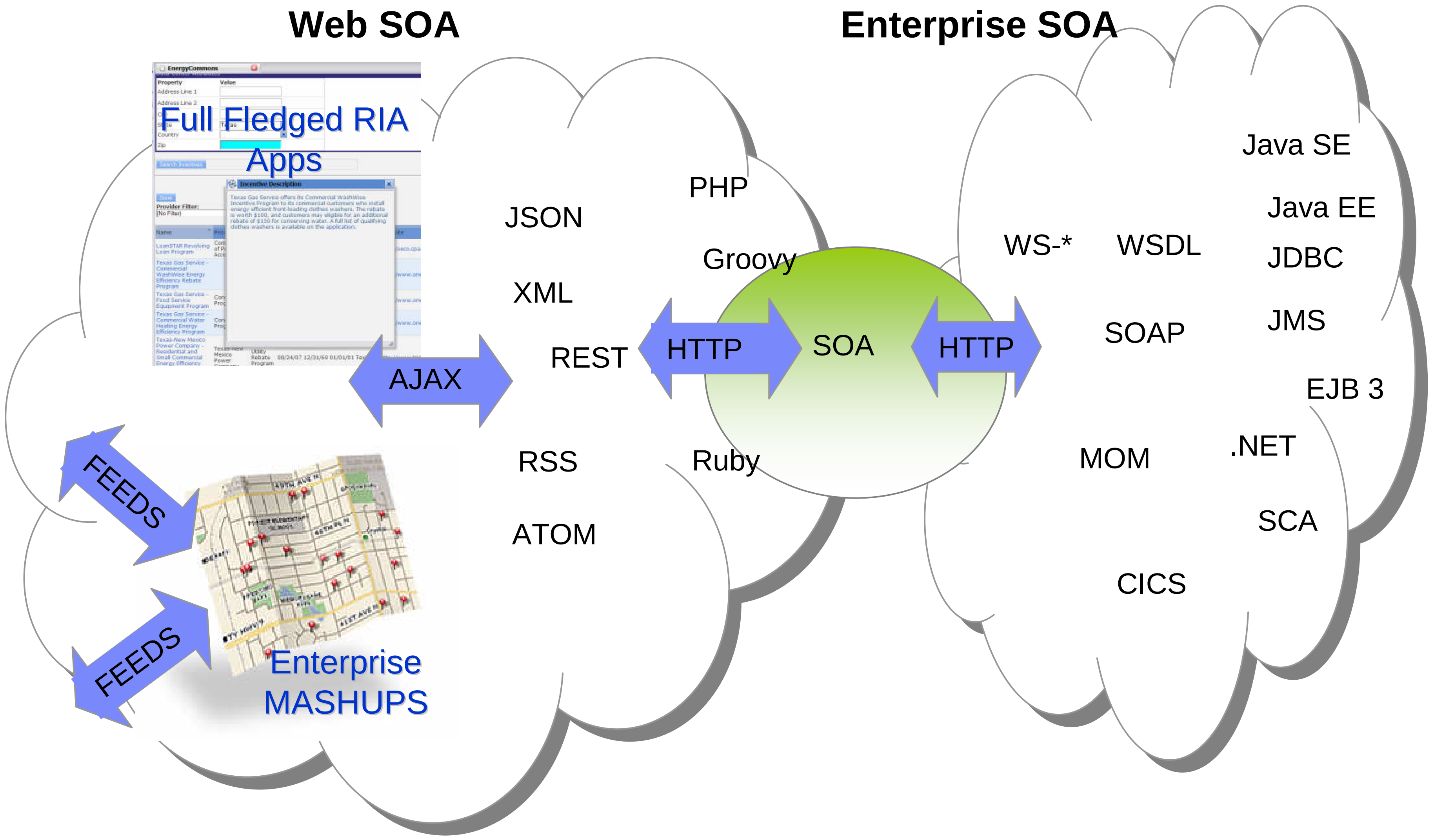
Verb = HTTP Action (GET, POST, PUT, DELETE)

Noun = the URI of the Service (the document)

Adjective = MIME type of the resulting document



Bridging Web SOA and Enterprise SOA



WebSphere sMash and REST

Resources on the Web

What are the URIs?

Which methods are supported at each URI?

What formats?

Resource	URI	Method	Representation	Description
Employee list	/resources/employee	GET	JSON (list)	List
		POST	JSON (employee)	Create
Employee	/resources/employee/{id}	GET	JSON (employee)	Retrieve
		PUT	JSON (employee)	Update
		DELETE		Delete

Representations

Employee

```
{
  "first_name“ : "Alice",
  "last_name“  : "Rogers",
  "location"   : "Seattle"
}
```

List of employees

```
[{
  "first_name“ : "Alice",
  "last_name“  : "Rogers",
  "location"   : "Seattle"
 },
 {
  "first_name“ : "Bill”,
  "last_name"  : "Stevens",
  "location"   : "Seattle“
}]
```

Resource Handlers in Zero

Basic event handlers for /resources/*

URI pattern	Method	Event	Description
/resources/collection	GET	list	List of all members
	POST	create	Create member
/resources/collection/{id}	GET	retrieve	Retrieve one member
	PUT	update	Replace member
	DELETE	delete	Delete member

Resource Handlers Example

app/resources/employee.groovy

```
def onList() {
  try {
    // Get configured DataManager for data access
    def data = zero.data.groovy.Manager.create('employee_db')

    // Retrieve employee records via Data Zero
    def result = data.queryArray('SELECT * FROM employees')
    request.view = 'JSON'
    request.json.output = result
    render()

  } catch (Exception e){
    if (e.getCause() instanceof java.sql.SQLException) {
      request.status = HttpURLConnection.HTTP_INTERNAL_ERROR
      request.view = 'error'
      request.error.message = 'The db may not have been initialized.'
      render()
    }
  }
} def onCreate() { ...
```

Resource Handlers Example

app/resources/employee.groovy (continued)

```
def onCreate() {
  // Convert entity to JSON object
  def emp = zero.json.Json.decode(request.input[])

  // Get configured DataManager for data access
  def data = zero.data.groovy.Manager.create('employee_db')

  // Insert employee record via Data Zero APIs
  data.update("""
    INSERT INTO employees
      (username, firstname, lastname, location, phonenumber)
    VALUES ($emp.username, $emp.firstname, $emp.lastname,
            $emp.location, $emp.phonenumber)
""")

  // Set a Location header with URI to the new record
  locationUri = getRequestedUri(false) + '/' + emp['username']
  request.headers.out.Location = locationUri
  request.status = HttpURLConnection.HTTP_NO_CONTENT
}
```

Renderers

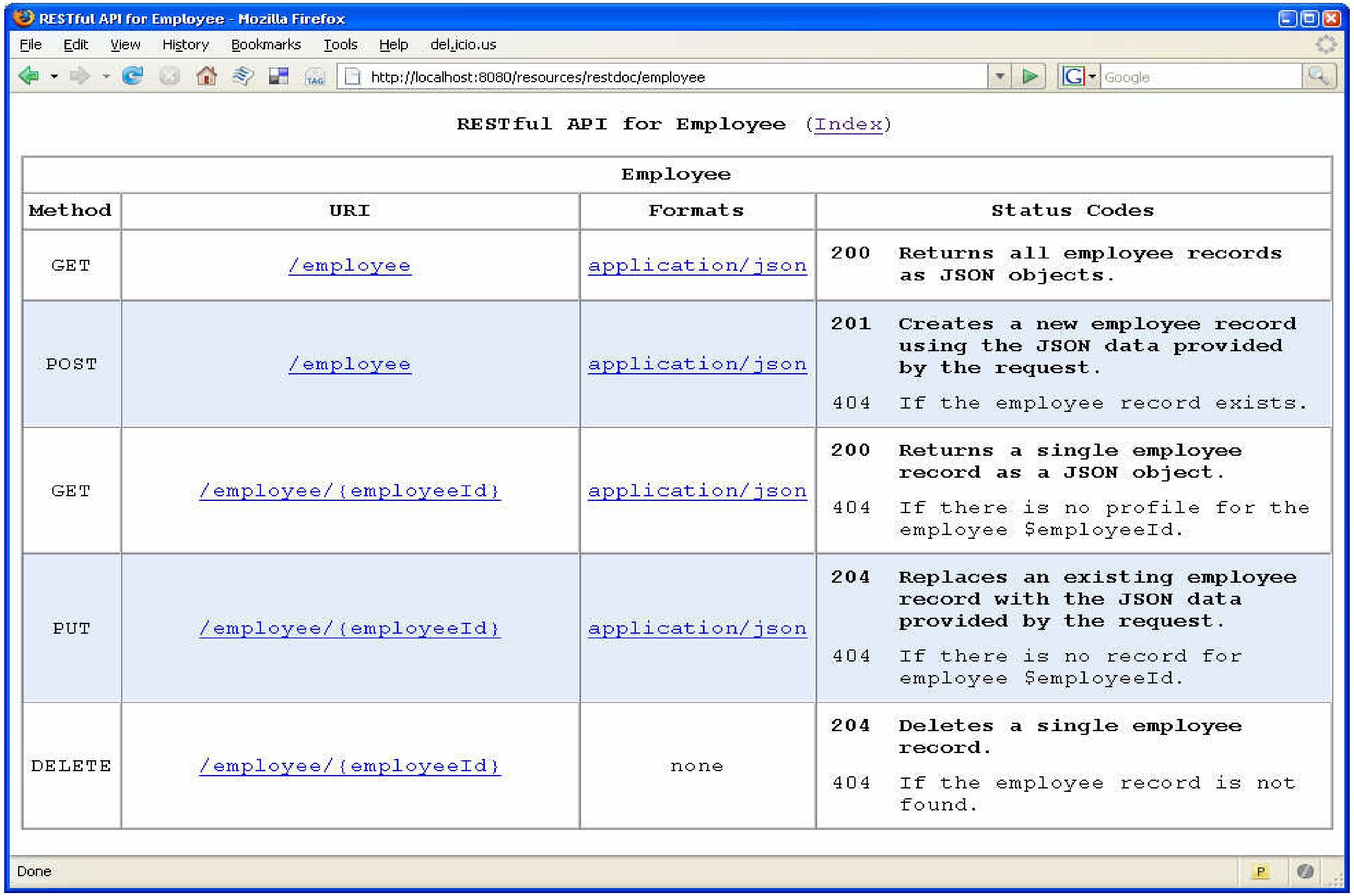
```
request.view = ‘JSON’

request.json.output = object

render()
```

JSON, XML

ATOM

Map	=> Atom entry
List<Map>	=> Atom feed

RESTdoc


RESTful API for Employee - Mozilla Firefox
File Edit View History Bookmarks Tools Help del.icio.us
http://localhost:8080/resources/restdoc/employee
Google
Done

RESTful API for Employee (Index)

Employee				
Method	**URI**	**Formats**	**Status Codes**	
GET	/employee	application/json	**200**	**Returns all employee records as JSON objects.**
POST	/employee	application/json	**201**	**Creates a new employee record using the JSON data provided by the request.**
			404	If the employee record exists.
GET	/employee/{employeeId}	application/json	**200**	**Returns a single employee record as a JSON object.**
			404	If there is no profile for the employee $employeeId.
PUT	/employee/{employeeId}	application/json	**204**	**Replaces an existing employee record with the JSON data provided by the request.**
			404	If there is no record for employee $employeeId.
DELETE	/employee/{employeeId}	none	**204**	**Deletes a single employee record.**
			404	If the employee record is not found.

An alternative: Zero Resource Model (ZRM)

Model application data

- Constrained set of APIs encourages a **RESTful** application architecture
- Data model that maps well into Atom feeds and JSON formats
- Robust framework for persistence, validation, and serialization

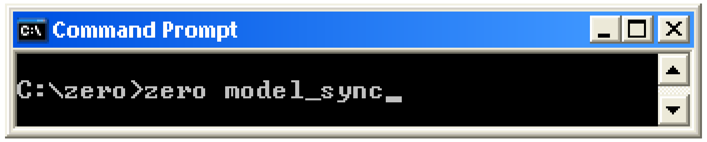
ZRM (continued)

app/models/employee.json

```
{
    "fields" : {
        "first_name": {"type":"string"},
        "last_name": {"type":"string"},
        "location": {"type":"string"}
    }
}
```

app/resources/employee.groovy

```
ZRM.delegate();
```


Command Prompt

```
C:\zero>zero model_sync_
```

app/models/fixtures/initial_data.json

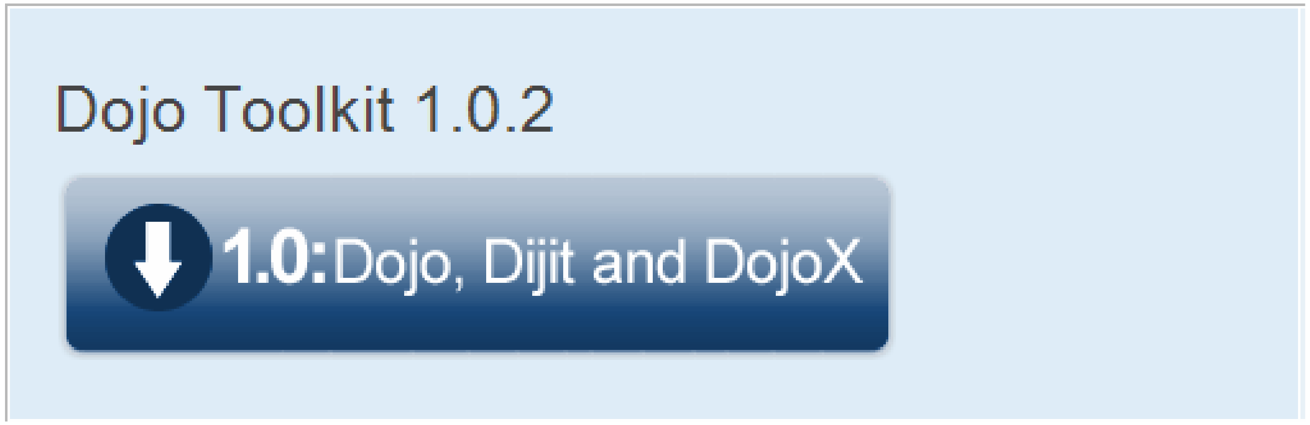
```
[
    {
        "type": "employee",
        "fields": {
                "first_name" :   "Alice",
                "last_name“  :   "Rogers",
                "location"   :   "Seattle"
        }
    },
    {
        "type": "employee",
        "fields": {
                "first_name" :   "Bill",
                "last_name"  :   "Stevens",
                "location"   :   "Seattle"
        }
    },
    {
        "type": "person",
        "fields": {
                "first_name" :   "Cathy",
                "last_name"  :   "Tomlin",
                "location"   :   "Boston"
        }
    }
]
```

WebSphere sMash and RIA

Dojo Broswer Toolkit

Dojo is an Open Source DHTML toolkit written in JavaScript. It builds on several contributed code bases.

- Provides Rich Set of Widgets
- Web UI Framework
- Rich Event handling System
- General Purpose HTML Libraries
- Several other utilities
  - Math, XML to JS parsing, etc…



Dojo Architecture

Base

The kernel of the toolkit wrapped into a **25k** js file (dojo.js). Base bootstraps the toolkit, includes AJAX utilities, class based inheritance, packaging system and more

Core

Provides addition facilities on top of the base for accessing data stores, effects such as wipes/slides, internationalization (i18n) and back-button handling among other things. Separate package keeps base small

Dijit

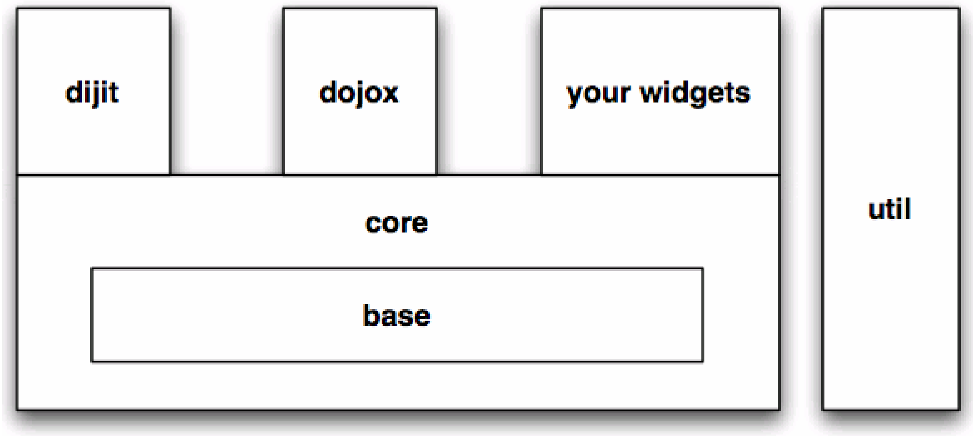
Shorthand for "Dojo widget". Could refer to a single Dojo widget (a dijit) or to the entire component of the toolkit containing all of Dojo's widgets (Dijit)

DojoX

“Dojo Extra" and contains features that stand a chance of one day migrating into Core, Dijit or even a new module. A great proving ground for new features while maintaining standards of core and base.

Util

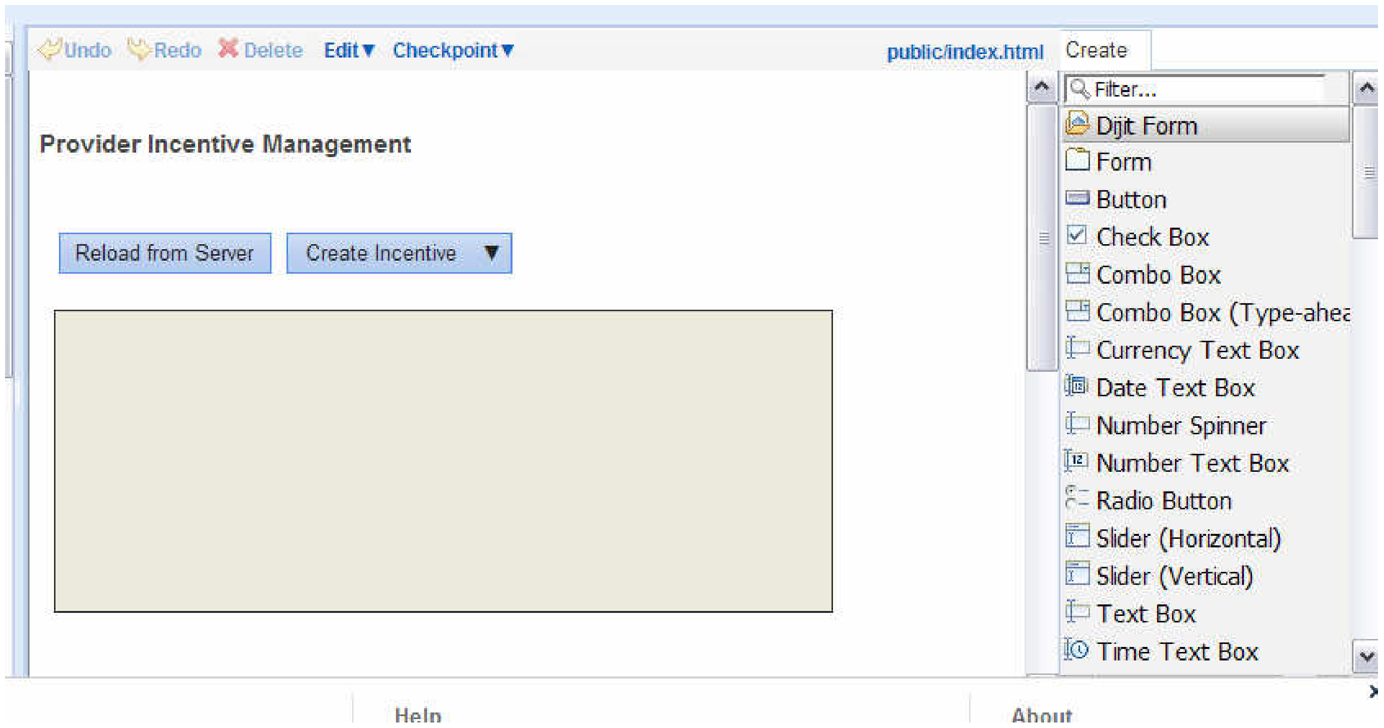
A collection of Dojo utilities (more later)



Web Based IDE Editor for Dojo

Dojo connections with Services through Wires.

Drag and Drop with Dojo Dijits.



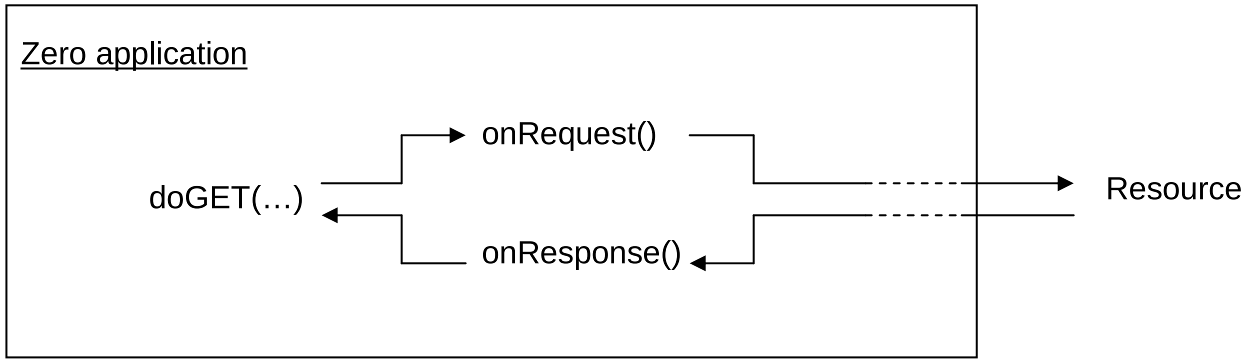
Connections

RESTful API to resources via a variety of protocols

- HTTP/S
- SMTP
- File
- JMS

In-process mediations

- SOAP
- Custom



WebSphere sMash and Flows

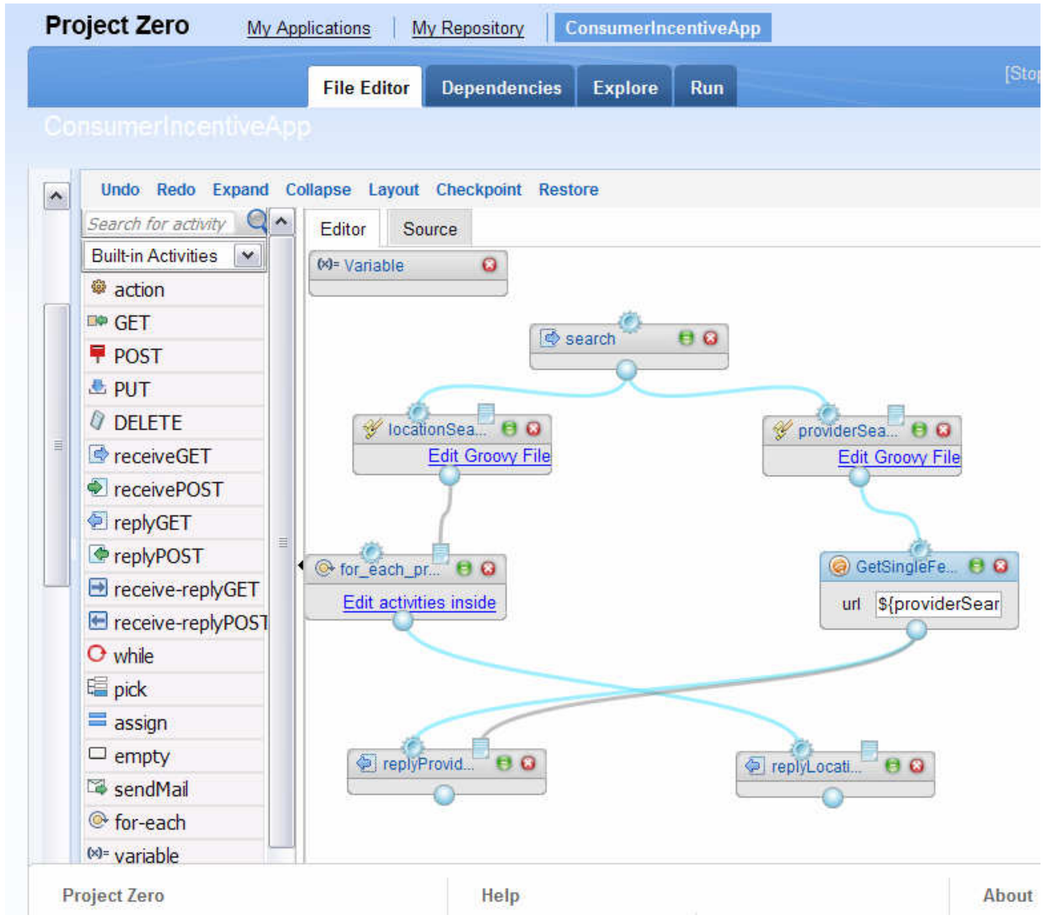
Assemble

Composition of applications by "wiring" REST services using the SPLICE flows. Incorporating both activity and data flows.

A solution may be rapidly assembled by combining existing feeds and services that enrich, sort, and filter data in a pipeline. Either visually or programmatically.

Configure templates to alter pipeline routes, log events along the pipeline, as well as transform data

Adapters to enhance integration with existing systems.



Questions & Answers