From Zero to sMash

Denise Hatzidakis
Perficient, Inc
Director, Chief Technologist
denise.hatzidakis@perficient.com
Web 2.0 Applications Use SOA

Serving New Markets with Specific Needs via Situational Applications

SOA is pervasive here

From dozens of markets of millions of users to millions of markets of dozens of people

Web 2.0 applications using SOA reach here

Sales analysis

Dashboards

Enterprise applications

IT created situational applications

User created situational applications

Number of Applications

Usage

Denise Hatzidakis – From Zero to sMash
Web 2.0 is...

- There are a lot of definitions... but in general it is...

A living term describing **changing trends in the use of World Wide Web technology and web design** that aims to enhance creativity, information sharing, collaboration and functionality of the web.
Web 2.0 is ...

- Origin of the term was Tim O’Reilly’s article: [http://www.oreil.lynet.com/lpt/a/6228](http://www.oreil.lynet.com/lpt/a/6228)

- It is about design patterns and business models for the next generation of software

- Technology isn’t the point – in many ways, the technologies involved are overly simplistic, inefficient, and not new

- The key point of Web 2.0 is the social factor: it is about how people and their actions make the software better
The evolving Web Platform

Web 1.0 was about connecting computers and making technology more efficient for computers.

Web 2.0 is about connecting people, and making technology efficient for people.

Web 2.0 changes the way in which businesses interact with its customers.

Web 2.0:
Is about communities and social networks
Builds contextual relationships and facilitates knowledge sharing
Is about people and the way they collaborate
Where did Web 2.0 start ...

- 2000 – Roy Fielding describes REST in his doctoral dissertation
- 2002 – Google Web API introduced*
- 2002 – Amazon Web Services offered via SOAP or REST
  ➢ As of April 2003, 85% of the traffic was REST**
- 2002-2004 eBay offers SOAP or REST Web Services
- 2005 – Google Maps
- 2005 – Jesse James Garrett coins the term AJAX***
- 2005 – Tim O’Reilly coins the term Web 2.0
# Web 2.0 Trends

<table>
<thead>
<tr>
<th><strong>Social networks</strong></th>
<th>Technology that allows users to leverage personal connections.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RSS</strong></td>
<td>An XML standard that lets users collect and read content feeds.</td>
</tr>
<tr>
<td><strong>Open source software</strong></td>
<td>Publicly available software that can be copied or modified without payment.</td>
</tr>
<tr>
<td><strong>Blogs</strong></td>
<td>Online diaries of text, photos, or other media.</td>
</tr>
<tr>
<td><strong>Search engines</strong></td>
<td>Services that find Web content based on user-specified criteria.</td>
</tr>
<tr>
<td><strong>User review portals</strong></td>
<td>Web portals that allow users to search for peer reviews on a product or service.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>LinkedIn</strong></th>
<th><strong>Facebook</strong></th>
<th><strong>Orkut</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MySpace</strong></td>
<td><strong>Friendster</strong></td>
<td><strong>MySpace</strong></td>
</tr>
<tr>
<td><strong>FeedBurner</strong></td>
<td><strong>Bloglines</strong></td>
<td><strong>Yahoo</strong></td>
</tr>
<tr>
<td><strong>Newsgator</strong></td>
<td><strong>Pluck</strong></td>
<td><strong>Google</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>P2P file sharing</strong></th>
<th>Sharing media files over a network powered by users who act as both client and server.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C2C eCommerce</strong></td>
<td>Buying and selling among consumers via the Net.</td>
</tr>
<tr>
<td><strong>Comparison shopping sites</strong></td>
<td>Sites that allow consumers to compare products or services.</td>
</tr>
<tr>
<td><strong>Podcasts</strong></td>
<td>Online audio or video that users can download to a device.</td>
</tr>
<tr>
<td><strong>Wikis/Collaboration software</strong></td>
<td>Shared publishing software or site that allows users to edit content.</td>
</tr>
<tr>
<td><strong>Tagging</strong></td>
<td>Metadata assigned to items like photos or Web pages to facilitate searching and sharing.</td>
</tr>
</tbody>
</table>

![Web 2.0 Trend Logos](image)
A Situational Application is
- rapidly created (<5mins) to address an immediate need of an individual or community
- informal (lacks product quality look and feel)

A Mashup can be a form of a Situation Application that
- is comprised of 2 or more disparate components that are bound together through content

A Mashboard can be a form of a Mashup that
- is typically personalized and unique for an individual yet configurable
Web 2.0 is...

- Although the term suggests a new version of the World Wide Web, it does not refer to an update to any technical specifications, but to changes in the ways software developers and end-users utilize the Web.

- Web 2.0 encourages lightweight business models enabled by syndication of content and of service and by ease of picking-up by early adopters.

- Via things like social-networking sites, video sharing sites, wikis, blogs, and folksonomies.
The Web 2.0 Hierarchy

Level-0
Applications work as well offline as online.
Examples → MapQuest, Yahoo! Local, and Google Maps (mapping-applications using contributions from users to advantage could rank as "level 2").

Level-1
applications operate offline but gain features online.
Examples → Writely (now Google Docs & Spreadsheets) and iTunes (because of its music-store portion

Level-2
Applications can operate offline but gain advantages from going online.
Example → Flickr, which benefits from its shared photo-database and from its community-generated tag database.

Level-3
Applications, the most "Web 2.0"-oriented, exist only on the Internet, deriving their effectiveness from the inter-human connections and from the network effects that Web 2.0 makes possible, and growing in effectiveness in proportion as people make more use of them.
Examples → eBay, Craigslist, Wikipedia, del.icio.us, Skype, dodgeball, and AdSense as examples.
## Technology Attributes of Web 2.0

<table>
<thead>
<tr>
<th>Rich User Experience</th>
<th>AJAX incorporating: XHTML and CSS, DOM, XML and XSLT;, XMLHttpRequest and JavaScript allowing information to be mashed up into new interactive portals.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lightweight Protocols</td>
<td><strong>XML or JSON data over HTTP</strong>, in a lightweight approach sometimes referred to as <strong>REST</strong> (Representational State Transfer) as an alternative to SOAP.</td>
</tr>
<tr>
<td>REST</td>
<td>&quot;<strong>DATA is the new HTML.</strong>&quot; Database management is a core competency of Web 2.0 companies.</td>
</tr>
<tr>
<td>Feeds</td>
<td><strong>RSS/ATOM</strong> allows someone to link not just to a page, but to subscribe to it, with notification every time that page changes.</td>
</tr>
<tr>
<td>Scripting</td>
<td>Scripting languages like <strong>PHP and Groovy</strong> allow for quick creation of web sites from existing data sources.</td>
</tr>
</tbody>
</table>
Java Runtimes Today

- Enterprise workloads influenced Java’s performance at any cost mentality
  - Benchmarks measure sustained throughput after a warm-up period

- Java’s original performance shortfalls hurt program design
  - Monolithic, long lived applications got best performance
  - JVMs got faster – at executing monolithic, long lived applications
  - A vicious cycle..

- Everything, including the kitchen sink programming model
  - Java is extremely capable, but at a footprint cost
The New Reality Runtime

- Web2.0 is the New Reality
  - Java needs to be optimized for the New Reality
  - Start-up time and memory footprint are key metrics

- Vision
  - Small initial download
    - Incremental future downloads as more function is required
  - Small memory footprint
    - Application footprint logically maps to complexity of application
  - OS process level isolation (1000’s of applications)
  - Strong support for Dynamic Languages
  - Universally available
What is Project Zero?

- Platform for building Web 2.0 applications
- REST based Programming model and runtime
- Scripting environment supporting PHP and Groovy
- Based on conventions
- Support for key technologies such as Feeds, AJAX ...
What is Project Zero

- Project Zero is the development and incubation community for IBM’s Web 2.0 Platform - 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
What is Project Zero

WebSphere sMash is
an Agile Web Application Platform

- Architectured around
  - Dynamic Scripting,
  - REST,
  - Rich Web Interfaces,
  - AJAX,
  - and Feeds.

- Optimized for
  - speed of development,
  - simple deployment,
  - and cost-effective operation.
Project Zero is....

The community development site for IBM WebSphere sMash

You can get WebSphere sMash DE (Developer Edition), the free download providing the tooling support and a stable runtime for testing and running applications, as well as check out our latest release under development (codename: Silverstone).
ProjectZero.org Contents

- Content
  - Documentation
  - Project information
  - Roadmaps
  - Design documents
  - Demos and Samples
  - iCal calendar
  - Future plans

- Source code (Subversion)

- Binary Downloads (approaching 600,000 downloads)

- Bug Tracking System (Bugzilla)

- Forum for interactive discussion
  - Help and Feedback for questions from users
  - Developer Alerts for new features and breaking changes
  - Thousands of Posts on 828 Topics to date

- Blogs
  - Development blog with interesting commentary, demos, and opinion
  - News blog for project announcements
www.projectzero.org

Anonymous Visitors can...
- Browse the site
- View Wiki content
- Read Forums
- Search the Bug Database
- Read Blogs
- Download Binary Drivers*

Registered Users can...
- Post to the Forum
- Submit Bug Reports
- Submit Feature Requests
- Comment on Blog Posts
- Access Source Code*

Focused on easy access
- Internet web site
- Free access to the platform

Focused on feedback
- Simple, free registration process

* Requires acceptance of an IBM license agreement
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
Is this an open source project?

- Project Zero is a commercial project
  - Developed in a very open way that many have found useful
  - Source code is available
  - A very transparent development process

- Does this mean IBM will not be pursuing new open source activities?
  - Of course not. IBM remains committed to open source.
  - This just happens to be one of IBM’s commercial projects.
Community Driven Commercial Development

Why does IBM think "community driven commercial development" is a good strategy?

- The community driven commercial development process is designed to open up IBM's approach to commercial software development by asking customers to weigh in on future product releases as they are being developed.

  - With regular milestones throughout the development of a release, users get the opportunity to see what is coming, provide feedback, and help set priorities.
  - Suggestions and decisions throughout the development cycle are logged in the public view for educational and historical purposes.
Community Driven Commercial Development

Why does IBM think "community driven commercial development" is a good strategy?

- As the Project Zero community grows, there will be a facility for community members to share reusable components (for example, libraries, widgets and services).
  - This is similar to the way Eclipse plug-ins work in that plug-ins extend the base Eclipse platform, but the plug-in itself is owned by the person or company that developed it.

- Someone commented that this approach is kind of like going into a restaurant and being able to see the chef prepare your meal. While you can not supply your own ingredients, you do have a say in the end result as it is being cooked.
Adoption

- Since Launch in July 2007
- Over 580,000 downloads
- Over 154,000 unique visitors to the website
- Over 2,400 referring websites
- Visitors from 179 countries
  - Singapore Ranked 13
  - Thailand Ranked 25
  - Malaysia Ranked 37
- 51% of all visitors return to the site
- Over 1500 registered users
- 630 registered forum users created 4885 posts on 1159 topics
- 800 External Blog Posts about sMash
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.
## WebSphere sMash Packaging

<table>
<thead>
<tr>
<th>Package Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WebSphere sMash</strong></td>
<td>Production version of the WebSphere sMash Platform. Standard IBM commercial license. Available through normal IBM channels.</td>
</tr>
<tr>
<td><strong>WebSphere sMash Reliable Transport Extensions</strong></td>
<td>Production version of the extended features in sMash related to messaging and reliable communications. Standard IBM commercial license. Available through normal IBM channels.</td>
</tr>
<tr>
<td><strong>WebSphere sMash Developer Edition (DE)</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>Project Zero</strong></td>
<td>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.</td>
</tr>
</tbody>
</table>
Project Zero concepts

- Java, PHP and Groovy
- Application-centric runtime
- Conventions based application directories
- Loose coupling with Events
- REST and Resources
- Share information using Global Context
WebSphere sMash innovations

CREATE
– Web-Oriented Programming Model

ASSEMBLE
– Simple Application and Service Assembly

EXECUTE
– “New Reality” Runtime
Create

- Simple Conventions to avoid excessive code and configuration

- Dynamic Scripting and Templates
  - PHP language syntax
  - Groovy (Java Language Syntax)
  - Java as System Programming Language

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

- Data Access using pureQuery

- Reactive Client integrated Dojo with Zero

- State externalized into a shared memory space (Global Context)

- State-less, Event-driven architecture

- Catalog of Services and Libraries providing useful building blocks
Core Application Concepts

- 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
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
<?php
    echo "Hi, I'm a PHP script!";
?>
Groovy support

- **Groovy...**
  - is an agile and **dynamic language** for the **Java Virtual Machine**
  - builds upon the strengths of Java but has **additional power features** inspired by languages like Python, Ruby and Smalltalk
  - makes **modern programming features** available to Java developers with **almost-zero learning curve**
  - supports **Domain-Specific Languages** and other compact syntax so your code becomes **easy to read and maintain**
  - makes writing shell and build scripts easy with its **powerful processing primitives**, OO abilities and an Ant DSL
  - increases developer productivity by **reducing scaffolding code** when developing web, GUI, database or console applications
  - **simplifies testing** by supporting unit testing and mocking out-of-the-box
    - seamlessly integrates with all existing Java objects and libraries
    - compiles straight to Java bytecode so you can use it anywhere you can use Java
Groovy

Standard Java (Java 5+)

for (String item : new String [ "Rod", "Carlos", "Chris" ]) {
  if (item.length() <= 4) {
    System.out.println(item);
  }
}

Groovy

["Rod", "Carlos", "Chris"].findAll{it.size() <= 4}.each{println it}
sMash Application Builder

- Browser-Based Development IDE
- Built as a sMash application
- Provides full development lifecycle for sMash 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

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
- Adapters to enhance integration with existing systems
Rapidly Expose Data
RESTfully

Robust framework for persistence, validation, and serialization

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

```python
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
```
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
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
Modular Architecture

- WebSphere sMash applications are based on a very small core
  - 5.4 MBytes (includes Groovy).
  - PHP adds additional 14.5 Mbytes (Size includes all 3 supported platforms)
  - 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

```xml
<dependencies>
  <dependency name="zero.core" org="zero" rev="[1.0.0.0, 2.0.0.0]"/>
  <dependency name="zero.php" org="zero" rev="[1.0.0.0, 2.0.0.0]"/>
</dependencies>
```
Runtime Characteristics

- **Instant On**
  - Application Available for Service in 1 sec
    - 1.078 seconds on my MacBook Pro
  - Application JVM starts in about 1 second
    - 1.332 seconds on my 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 ~348 Kbytes
  - Running Application JVM ~30 Mbytes TODAY

- **Supported on “stock” JVM**
  - IBM, Sun, Mac, etc. – Any JSE 5 or 6 JVM
Runtime Model

Initial Request

Following Requests

Port

ZSO Detects Request

ZSO Creates Socket

Zero Socket Opener (ZSO)

Zero Application

ZSO waits for next request

Application runs until Idle timeout or max requests Reached. Then JVM terminates cleanly

ZSO starts application And transfers port ownership

Following requests go straight to application
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**
```java
def onGET()
{
    println "Hello World"
}
```

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

**Java**
```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 restartability 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 Zones

**Zones in the Global Context represent different data lifecycles**

<table>
<thead>
<tr>
<th>Zone</th>
<th>Scope</th>
<th>Automatic Recycle</th>
<th>User Initiated Restart</th>
<th>User Modified Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>App</td>
<td>Application</td>
<td>State Preserved</td>
<td>State Discarded</td>
<td>preserved</td>
</tr>
<tr>
<td>Request</td>
<td>Request / Thread</td>
<td>State Discarded</td>
<td>State Discarded</td>
<td>discarded</td>
</tr>
<tr>
<td>Event</td>
<td>Event / Thread</td>
<td>State Discarded</td>
<td>State Discarded</td>
<td>discarded</td>
</tr>
<tr>
<td>Config</td>
<td>Application</td>
<td>State reloaded from config files</td>
<td>State reloaded from config files</td>
<td>discarded</td>
</tr>
<tr>
<td>User</td>
<td>Session denoted by zsessionid</td>
<td>State Preserved</td>
<td>State Discarded</td>
<td>preserved</td>
</tr>
<tr>
<td>Tmp</td>
<td>Application</td>
<td>State Discarded</td>
<td>State Discarded</td>
<td>discarded</td>
</tr>
<tr>
<td>Storage</td>
<td>Application</td>
<td>State Preserved</td>
<td>State preserved</td>
<td>preserved</td>
</tr>
</tbody>
</table>
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

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

- SampleZeroApp
  - java
    - myapp
    - Impl.java
  - public
    - foo
      - bar.gjt
    - images
      - hello.gif
      - index.groovy
  - app/resources
    - employees.bnd
    - employees.groovy
  - app/errors
    - error.html
    - error404.groovy
    - error500.groovy
  - app/scripts
    - error.groovy
    - index.groovy
  - app/views
    - utils.groovy
    - EmployeeEditor.gjt
    - login.gjt
  - JRE System Library (JVM 1.5.0)
    - Zero Local Libraries
    - Zero Resolved Libraries
  - java
    - app
    - classes
      - config
        - ivy.xml
        - logging.properties
        - zero.config
      - lib
  - config
    - config.ivy.xml
    - config/logging.properties
    - config/zero.config
  - public
    - The Web accessible root folder of the application.
      - http://localhost:(port)/
      - The public folder can contain static content (for example HTML, images, CSS, or JS), scripts (.groovy and .jxt), and templates (.gt). The file serving section contains more details.
Virtualized Directories

- WebSphere sMash 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.
### Configuration – zero.config

#### zero.config
- Processed at the start of a Zero application
- Organized into "stanzas" of related key/value pairs.
- Stanzas are associated with directives, such as
  - "store to the Global Context"
  - "include another configuration file."

#### Sample Configuration

```plaintext
# 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 = "/config/property/myPrefix/bat"

# Include
@include "/config/dependencies/zero.core/config/security/form.config"
```

{ "formLoginPage" : "/login" }
Rendering

- Rendering
  - Direct
  - Indirect

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

```groovy
def onGet() {
    println "Hello World"
}
```

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

```php
<?php
    request.view = 'x/hello.php';
    render();
?>
```

Custom Groovy Templates

Custom PHP Templates

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

JSON Render

Error Render
Renders

- **View Renderer**
  - The View renderer serves a file from the app/views virtualized directory.
  - Files are processed in a similar fashion to standard file serving in Zero:
    - executable files are "invoked" and static files are served directly.
  - To use this renderer,
    - set /request/view to the file path relative to app/views.

- **Error Renderer**
  - The Error renderer is tailored for error reporting and uses scripts from the app/errors virtualized directory:
    - zero.core includes a set of basic error pages;
    - zero.core.webtools includes error pages that are more descriptive for developers.
  - To use this renderer,
    - set /request/view to "error" and
    - set /request/error/page to the path to the script relative to app/errors.

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

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

- JSON Renderer
  - The JSON renderer serializes Java objects to the outputstream as JSON representations.
    - Details about this serialization may be found on the JSON support documentation.

- To use this renderer,
  - set /request/view to "JSON" and
  - set /request/json/output to the object to be serialized.

- The renderer will set the Content-Type response header to application/json, if not already set.
**Renders**

- **XML Renderer**
  - The Extensible Markup Language (XML) renderer serializes Java objects to the output stream as XML.
  
  - To use this renderer,
    - set /request/view to "XML" and
    - set /request/xml/output to the object to be serialized.
  
  - Note, that if the object to be serialized is a subtype of org.w3c.dom.Node interface (such as Document or Element), the resulting XML will be the contents of the Node object (which is what you want), not a XML rendering of the Node Java object.

- You can optionally
  - set /request/xml/rootElement to the name of the XML tag that will form the root element of the produced XML document.
    - If the /request/xml/rootElement value is not specified then the lower cased simple name of the Java Object will be used as the root element name.
  
  - set /request/xml/stylesheets to the name of an XSL stylesheet located in the app/views folder of the project (or its dependencies).

- This will cause the stylesheets to be applied to the resulting XML document and the XSL output to be sent to the client. The value of the /request/xml/stylesheets attribute is a String relative path to the XSL document within the app/views folder.
REST and Resources

REpresentational State Transfer (Roy Fielding)

- Everything is modeled as a Resource
- Every resource is identified by an address (URI)
- Resources have state (representation)
- HTTP is used to transfer state to networked application

- HTTP verbs operate on the resource
  - **GET** → retrieves a resource’s state representation
  - **POST** → Updates resource (or other processing)
  - **DELETE** → deletes resource
  - **PUT** → Creates / updates resource state

![Diagram showing server, resource, state, and HTTP methods](image)
Restful Resources

- **RESTful Design**
  - Collection Model
  - Action can be taken on the entire collection or a specified member of the collection
  - URI and HTTP method define the resource request

- **REST and WebSphere sMash**
  - WebSphere sMash supports
  - URI and HTTP method define the collection resource model
  - Each script in the `<apphome>/app/resources` directory represents a resource handler
  - URL convention for interacting with resources based on
    `/resources/<collectionName>[/<memberID>[/<pathInfo>]]`
    where the actions are defined as follows:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Get</th>
<th>Put</th>
<th>Post</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collection</td>
<td>list</td>
<td>putCollection</td>
<td>create</td>
<td>deleteCollection</td>
</tr>
<tr>
<td>Member</td>
<td>retrieve</td>
<td>update</td>
<td>postMember</td>
<td>delete</td>
</tr>
</tbody>
</table>
Available Modules

- There are 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
How much does it cost?

- The code available on the projectzero.org site is free, but non-warranted and not supported.
  - It is best used for development and small production deployments. The license agreement at www.projectzero.org has further details.

- The commercial product, WebSphere sMash, is available for larger, production deployments, and comes with world class support from IBM.
  - You can learn more about WebSphere sMash by visiting the WebSphere sMash product page.
Past, Present, Future


- IBM created the initial seeds of this project internally and will continue to drive the development of the technology with feedback and contributions from the community as described in some of the following sections.

- In the Future, the Project Zero community and site will continue to serve as an incubator for technologies that speed the development of useful web applications.

- The code that resides here will continue to be available as a non-commercial entity.

- Visitors will have access to stable builds (WebSphere sMash Developer Edition), tooling, and the latest features that are not yet available in the commercial WebSphere sMash product.