COMPARING JVM WEB FRAMEWORKS

Matt Raible
http://raibledesigns.com
Introductions

› Your experience with web development?

› Your experience with Java EE development?

› What do you want to get from this session?

› Experience with Grails, GWT, Rails, Spring MVC, Wicket, Tapestry or Play?
Who is Matt Raible?

Father, Skier, Cyclist

Web Framework Connoisseur

Founder of AppFuse

Blogger on raibledesigns.com
Session Agenda

- The Problem with Web Frameworks
- The Candidates
- Comparison Points
- The Matrix
- Conclusion
- Q and A
The Problem
The Real Problem
The Real Problem
The Real Problem
How do you choose?
Eliminate, Don’t Include
I've got a new way for you to build web apps!
... while I'd *love* to see life made simpler for Java web developers, and a lot of the things happening in Struts2 are going that way -- it won't be me doing it.

I've gone over to the dark side :-) and **much prefer to develop in Rails** -- for the conciseness mentioned above, but also **because I don't ever have to do a "build" or "deploy" step** during my development cycle any more. But you guys and gals need to be reminded that *this* is the kind of thing you are competing against if you expect to attract Rails developers ... or to avoid even more "previously Java web developer" defectors like me :-).

-- Craig McClanahan, 10/23/2007

http://markmail.org/thread/qfb5sekad33eobh2
James Gosling on JSF

http://www.youtube.com/watch?v=9ei-rbULWoA&amp;t=47m
2010: Comparison Points

- Developer Productivity
- Developer Perception
- Learning Curve
- Project Health
- Developer Availability
- Job Trends
2010: Comparison Points

- Templating
- Components
- Ajax
- Plugins or Add-Ons
- Scalability
- Testing Support
2010: Comparison Points

- i18n and l10n
- Validation
- Multi-language Support (Groovy / Scala)
- Quality of Documentation/Tutorials
- Books Published
- REST Support (client and server)
2010: Comparison Points

- Mobile / iPhone Support
- Degree of Risk
# Comparison Matrix

Matrix Results

<table>
<thead>
<tr>
<th>Framework</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grails</td>
<td>17.5</td>
</tr>
<tr>
<td>Spring</td>
<td>17</td>
</tr>
<tr>
<td>Rails</td>
<td>17</td>
</tr>
<tr>
<td>GWT</td>
<td>17</td>
</tr>
<tr>
<td>Vaadin</td>
<td>15.5</td>
</tr>
<tr>
<td>Wicket</td>
<td>15</td>
</tr>
<tr>
<td>Tapestry</td>
<td>15</td>
</tr>
<tr>
<td>Struts 2</td>
<td>14.5</td>
</tr>
<tr>
<td>Stripes</td>
<td>14</td>
</tr>
<tr>
<td>Play</td>
<td>14</td>
</tr>
<tr>
<td>Flex</td>
<td>13.5</td>
</tr>
<tr>
<td>JSF</td>
<td>13.5</td>
</tr>
<tr>
<td>Lift</td>
<td>11.5</td>
</tr>
</tbody>
</table>
Matrix Results

- Grails (17.5)
- GWT (17)
- Ruby on Rails (17)
- Spring MVC (17)
- Vaadin (15.5)
- Tapestry and Wicket (15)
# Weighted Matrix

<table>
<thead>
<tr>
<th>Weight</th>
<th>Criteria</th>
<th>Struts 2</th>
<th>Spring MVC</th>
<th>Wicket</th>
<th>JSF</th>
<th>Tapestry</th>
<th>Stripes</th>
<th>CWT</th>
<th>Grails</th>
<th>Rails</th>
<th>Flex</th>
<th>Vaadin</th>
<th>Lift</th>
<th>Play</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Developer Productivity</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>10.00</td>
<td>5.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>5.00</td>
<td>10.00</td>
<td>5.00</td>
<td>10.00</td>
</tr>
<tr>
<td>0</td>
<td>Developer Perception</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>Learning Curve</td>
<td>5.00</td>
<td>5.00</td>
<td>2.50</td>
<td>2.50</td>
<td>2.50</td>
<td>2.50</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>2.50</td>
<td>5.00</td>
</tr>
<tr>
<td>5</td>
<td>Project Health</td>
<td>2.50</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>2.50</td>
<td>2.50</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>2.50</td>
<td>5.00</td>
</tr>
<tr>
<td>5</td>
<td>Developer Availability</td>
<td>2.50</td>
<td>5.00</td>
<td>2.50</td>
<td>5.00</td>
<td>5.00</td>
<td>2.50</td>
<td>5.00</td>
<td>2.50</td>
<td>5.00</td>
<td>2.50</td>
<td>0.00</td>
<td>2.50</td>
<td>0.00</td>
</tr>
<tr>
<td>0</td>
<td>Job Trends</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0</td>
<td>Templating</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0</td>
<td>Components</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>Ajax</td>
<td>2.50</td>
<td>5.00</td>
<td>2.50</td>
<td>2.50</td>
<td>2.50</td>
<td>2.50</td>
<td>5.00</td>
<td>2.50</td>
<td>2.50</td>
<td>5.00</td>
<td>5.00</td>
<td>2.50</td>
<td>5.00</td>
</tr>
<tr>
<td>5</td>
<td>Plugins or Add-Ons</td>
<td>2.50</td>
<td>0.00</td>
<td>5.00</td>
<td>5.00</td>
<td>2.50</td>
<td>0.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>2.50</td>
<td>5.00</td>
</tr>
<tr>
<td>10</td>
<td>Scalability</td>
<td>10.00</td>
<td>10.00</td>
<td>5.00</td>
<td>5.00</td>
<td>10.00</td>
<td>10.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>10.00</td>
<td>10.00</td>
</tr>
<tr>
<td>10</td>
<td>Testing</td>
<td>10.00</td>
<td>10.00</td>
<td>5.00</td>
<td>5.00</td>
<td>10.00</td>
<td>10.00</td>
<td>5.00</td>
<td>10.00</td>
<td>5.00</td>
<td>0.00</td>
<td>5.00</td>
<td>5.00</td>
<td>10.00</td>
</tr>
<tr>
<td>0</td>
<td>i18n and i10n</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>Validation</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>2.50</td>
<td>2.50</td>
</tr>
<tr>
<td>10</td>
<td>Multi-language Support (Groovy / Scala)</td>
<td>5.00</td>
<td>5.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>0.00</td>
<td>10.00</td>
<td>0.00</td>
<td>0.00</td>
<td>10.00</td>
<td>0.00</td>
<td>5.00</td>
</tr>
<tr>
<td>10</td>
<td>Quality of Documentation/Tutorials</td>
<td>5.00</td>
<td>10.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
</tr>
<tr>
<td>0</td>
<td>Books Published</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>10</td>
<td>REST Support (client and server)</td>
<td>5.00</td>
<td>10.00</td>
<td>5.00</td>
<td>0.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>10.00</td>
<td>10.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>10</td>
<td>Mobile / iPhone Support</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
<td>5.00</td>
<td>10.00</td>
<td>10.00</td>
<td>10.00</td>
</tr>
<tr>
<td>0</td>
<td>Degree of Risk</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**Weighted Totals**: 70 65 67.5 62.5 75 77.5 80 90 82.5 50 82.5 62.5 62.5
Weighted Results

- Grails (90)
- Spring MVC (85)
- Ruby on Rails (82.5)
- Vaadin (82.5)
- Play (82.5)
- GWT (80)
Fighting for 5th

- Top at Devoxx 2010
  - GWT
  - Rails
  - Spring MVC
  - Grails
  - Wicket / Struts 2
Fighting for 5th

- Grails
- GWT
- Rails
- Spring MVC
- Tapestry / Vaadin

Top at Rich Web Experience 2010
Fighting for 5th

- Grails
- GWT
- Rails
- Spring MVC
- Vaadin

Top at TheServerSide Java Symposium 2010
Ratings Logic

- Developer Productivity
- Developer Perception
- Learning Curve
- Project Health
- Developer Availability
- Job Trends
Ratings Logic

- Templating
- Components
- Ajax
- Plugins or Add-Ons
- Scalability
- Testing Support
Ratings Logic

- i18n and l10n
- Validation
- Multi-language Support (Groovy / Scala)
- Quality of Documentation/Tutorials
- Books Published
- REST Support (client and server)
Ratings Logic

- Mobile / iPhone Support
- Degree of Risk

http://raibledesigns.com/rd/entry/how_i_calculated_ratings_for
David Pollack's Lift Ratings

- Developer Productivity: Lift gets a 11, Rails gets a 5, most Java-based frameworks get a 1 or less.
- Developer Perception: Every web framework gets a 1.
- Learning Curve: Lift gets a 2.
- Job Trends, yep, it's zero.

* Matt's scale is 0-1 and my ratings are on Matt's scale, except mine goes to 11.

http://lift.la/my-take-on-matt-raibles-spreadsheet
Peter Thomas's Perfbench

- Seam / JSF vs. Wicket Performance Comparison
  - January 2009: Seam 2.1.1 and Wicket 1.3.5
  - Average page response time in milliseconds

<table>
<thead>
<tr>
<th>action</th>
<th>Seam / JSF</th>
<th>Wicket</th>
</tr>
</thead>
<tbody>
<tr>
<td>get login</td>
<td>569</td>
<td>154</td>
</tr>
<tr>
<td>post login</td>
<td>392</td>
<td>303</td>
</tr>
<tr>
<td>ajax post search</td>
<td>35</td>
<td>15</td>
</tr>
<tr>
<td>get view hotel</td>
<td>19</td>
<td>8</td>
</tr>
<tr>
<td>post book hotel</td>
<td>28</td>
<td>28</td>
</tr>
<tr>
<td>ajax post cc number</td>
<td>23</td>
<td>4</td>
</tr>
<tr>
<td>ajax post cc name</td>
<td>18</td>
<td>3</td>
</tr>
<tr>
<td>post booking details</td>
<td>16</td>
<td>9</td>
</tr>
<tr>
<td>post confirm booking</td>
<td>33</td>
<td>29</td>
</tr>
<tr>
<td>get logout</td>
<td>15</td>
<td>8</td>
</tr>
</tbody>
</table>
On the Seam / JSF side, the 20 sessions each take up about 800 KB adding up to around 16 MB total. On the Wicket side the 20 sessions add up to around 1.5 MB.

+ Lots of banter between Peter and Seam developers @ http://bit.ly/3X50Gc
Peter’s Observations:

- Grails was far more productive than Tapestry 5.
- Grails still has some ways to go in terms of performance.
- Overall, Wicket is fastest, with Tapestry coming a close second.
- Wicket takes up the least amount of heap.
- Session usage of the Seam + JSF combination is significantly higher compared to the rest.
World Wide Wait - Devoxx

http://www.parleys.com/d/2942
### Statistics

<table>
<thead>
<tr>
<th>Library</th>
<th>LOC</th>
<th>Other LOC</th>
<th>Ext</th>
<th>WAR</th>
<th>WAR Extensions</th>
<th>HTML</th>
<th>Overview Page Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mule</td>
<td>332</td>
<td>5</td>
<td>0</td>
<td>14.7</td>
<td>57353</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JSF</td>
<td>121</td>
<td>7</td>
<td>0</td>
<td>15.4</td>
<td>47327</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring</td>
<td>137</td>
<td>4</td>
<td>0</td>
<td>17.4</td>
<td>24781</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GWT-P</td>
<td>637</td>
<td>9</td>
<td>0</td>
<td>35.6</td>
<td>14901*</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* only JSON data
> 16 GB of data
>700 hours of test runs
World Wide Wait - Devoxx

Cost of Scale

6.658 $ 9.986 $ 49.932 $ 59.918 $ 103.193 $

G  spring  JSF  MyFaces
HybridJava beats JSF, Wicket and Spring MVC by performance.
HybridJava is a Java™-based server-side component-oriented MVC technology for programming dynamic web content. Most of existing frameworks claim to support "components", but what they really mean is often one or several of the following:

- components of page Model
- components of page Controller
- components of page View

For instance, a popular book on Struts "Programming Jakarta Struts" has chapters 5-7 titled "Struts Controller Components", "Struts Model Components", and "Struts View Components". Those three types of components are notably uncorrelated, so making them work together takes quite an effort. This situation did not change much with introduction of Struts 2.

In truly component-oriented frameworks (Wicket, Tapestry, Click and HybridJava) each component follows the MVC paradigm independently, encapsulating its own Model, its own Controller, and its own View within a single entity.

Historically, component-oriented approach flourished in development of desktop applications UI. Wicket, Tapestry and Click openly admit to their inheritance of Visual Basic. The core of VB approach of constructing UI from components is having objects of a programming language organized in a data structure isomorphic to UI. Recursive visiting of this structure makes nodes add some output to the screen. Objects representing buttons are designed to fire events as traditional method calls up along the tree and so on.

With invention of markup languages it became possible to conveniently depict the structure of UI using tags without doing any programming. Wicket, Click and Tapestry, however, still use a programmatically built structure. In particular, Wicket is overly redundant at that. To add a Wicket subcomponent means to add a Java class node, depicted both in mark-up as well as in configuration. HybridJava technology finally moves the task of constructing pages and components completely to the mark-up area, eliminating the need for programming and configuring.
Pros and Cons
Grails

- Pros
  - Easy dynamic language transition for Java Developers
  - Groovy
  - Plugins for all types of applications

- Cons
  - Groovy learning targets Java Developers
  - Stack traces are horrendous
  - Knowledge of underlying frameworks not required, but helpful
GWT

- **Pros**
  - Write Java => Produces Optimized JavaScript
  - Easy to learn and develop with standard Java Tools
  - Vibrant Community

- **Cons**
  - You have to know Java
  - Slow to compile, difficult to test
  - More like a JSP Tag Library than a web framework
Ruby on Rails

- **Pros**
  - Easy to learn and understand for Web Developers
  - Lots and lots of documentation
  - Passionate Community

- **Cons**
  - Slightly less performant by default
  - Dynamic language means more tests
  - Development Tools and Debugging
Spring MVC

**Pros**
- Easy Configuration with Annotations and Conventions
- Integrates with many view options seamlessly: JSP/JSTL, Tiles, FreeMarker, Excel, PDF, JSON
- Excellent REST Support

**Cons**
- Instant reload not built-in, need JRebel or Spring Roo
- No open development process, need to be SpringSource
- Ajax requires 3rd-party library (can be a good thing!)
Vaadin

› **Pros**
  - Uses GWT API for developing view
  - Vibrant Community and company backing
  - Excellent Themes and Layouts support

› **Cons**
  - Large memory footprint, state stored in session
  - Sketchy (?) because backed by a commercial organization
  - Joonas keeps telling me my cons are wrong
Wicket

Pros
- Great for Java Developers
- Tight binding between pages and views
- Active community - support from creators

Cons
- No Jobs or Developers
- Stateful by default
- HTML Templates live next to Java code by default
Tapestry

- **Pros**
  - Live Class Reloading
  - Development emphasis on performance and scalability
  - Excellent Exception Reporting

- **Cons**
  - No Jobs
  - Prototype baked in for JS Library
  - Annotations vs. Conventions
Pretty Graphs
LOC in AppFuse Light

- JSF
- Spring
- Stripes
- Struts 2
- Tapestry 5
- Wicket

JavaScript XML CSS Java
Jobs on Dice (World)
LinkedIn Skills (World)
Pretty Graphs

Google Trends

- Scale is based on the average worldwide traffic of grails in all years. Learn more
- An improvement to our geographical assignment was applied retroactively from 1/1/2011. Learn more

<table>
<thead>
<tr>
<th>Search Term</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>grails</td>
<td>1.00</td>
</tr>
<tr>
<td>gwt</td>
<td>2.85</td>
</tr>
<tr>
<td>ruby on rails</td>
<td>3.75</td>
</tr>
<tr>
<td>spring mvc</td>
<td>0.60</td>
</tr>
<tr>
<td>vaadin</td>
<td>0.15</td>
</tr>
</tbody>
</table>

Showing only first 5 terms

Search Volume index

© 2012 Raible Designs
Pretty Graphs

Grails, GWT, Ruby on Rails, Spring MVC, Vaadin, Java Server Faces Job Trends

Job Trends from Indeed.com

Scale: Absolute - Relative

Email to a friend
Post on your blog/website

Top Job Trends
1. HTML5
2. MongoDB
3. iOS
4. Android
5. Mobile app
6. Puppet
7. Hadoop
8. jQuery
9. PaaS
10. Social Media

Indeed.com searches millions of jobs from thousands of job sites. This job trends graph shows the percentage of jobs we find that contain your search terms.
Pretty Graphs

Grails, GWT, Ruby on Rails, Spring MVC, Vaadin, Java Server Faces Job Trends

Job Trends from Indeed.com

Scale: Absolute - Relative

Email to a friend
Post on your blog/website

Top Job Trends
1. HTML5
2. MongoDB
3. iOS
4. Android
5. Mobile app
6. Puppet
7. Hadoop
8. jQuery
9. PaaS
10. Social Media

Indeed.com searches millions of jobs from thousands of job sites. This job trends graph shows relative growth for jobs we find matching your search terms.
Pretty Graphs

Vaadin, Wicket, Tapestry, Play Framework Job Trends

Scale: Absolute - Relative

Top Job Trends
1. HTML5
2. MongoDB
3. iOS
4. Android
5. Mobile app
6. Puppet
7. Hadoop
8. jQuery
9. PaaS
Grails, GWT, Ruby on Rails, Java Struts Job Trends

Job Trends from Indeed.com
- Grails
- GWT
- Ruby on Rails
- Java Struts

Scale: Absolute - Relative

Indeed.com searches millions of jobs from thousands of job sites. This job trends graph shows the percentage of jobs we find that contain your search terms.
Grails, GWT, Ruby on Rails, Java Struts Job Trends

Indeed.com searches millions of jobs from thousands of job sites. This job trends graph shows the percentage of jobs we find that contain your search terms.
Pretty Graphs

Grails, GWT, Ruby on Rails, Java Struts, PHP Job Trends

Scale: Absolute - Relative

Job Trends from Indeed.com

Grails - GWT - Ruby on Rails - Java Struts - PHP

Percentage of Matching Job Postings

Indead.com searches millions of jobs from thousands of job sites. This job trends graph shows the percentage of jobs we find that contain your search terms.

Top Job Trends
1. HTML5
2. MongoDB
3. iOS
4. Android
5. Mobile app
6. Puppet
7. Hadoop
8. jQuery
9. PaaS
10. Social Media

Email to a friend

Post on your blog/website

© 2012 Raible Designs
ASP.NET, Ruby on Rails, Java Struts, PHP Job Trends

Top Job Trends
1. HTML5
2. MongoDB
3. iOS
4. Android
5. Mobile app
6. Puppet
7. Hadoop
8. jQuery
9. PaaS
10. Social Media

Indeed.com searches millions of jobs from thousands of job sites. This job trends graph shows the percentage of jobs we find that contain your search terms.
Pretty Graphs

.NET, Ruby, Java, PHP, Python, JavaScript Job Trends

Job Trends from Indeed.com

Scale: Absolute - Relative

Top Job Trends
1. HTML5
2. MongoDB
3. iOS
4. Android
5. Mobile app
6. Puppet
7. Hadoop
8. jQuery
9. PaaS
10. Social Media

Indeed.com searches millions of jobs from thousands of job sites. This job trends graph shows the percentage of jobs we find that contain your search terms.
Mailing List Traffic

* Spring MVC and Vaadin use Forums, which don’t provide this data.
Books on Amazon

- Rails
- Spring MVC
- GWT
- Grails
- Wicket
- Tapestry 5
- Vaadin

0 75 150 225 300
2011 Releases

As of August 24, 2011

- Grails
- GWT
- Rails
- Spring MVC
- Vaadin
- Tapestry
- Wicket

© 2012 Raible Designs
StackOverflow

Tagged Questions (August 24, 2011)

- Rails: 50,000
- Vaadin: 37,500
- Grails: 25,000
- Other: 12,500

© 2012 Raible Designs
Tagged Questions (August 24, 2011)

- Spring
- JSF
- GWT
- Grails
- Wicket
StackOverflow

Tagged Questions (August 24, 2011)

Scala
Groovy
Clojure
JRuby
Framework Popularity

Source: ZeroTurnaround's Java EE Productivity Report 2010
What we need is...
Modern Principles in Web Development

By Rich Manslang, Developer Advocate
About Developer
On January 18, 2012

I've been kickstarting a bunch of small web apps lately. It seems like every time I start a new project, there's always something new that causes me to adjust my development principles. I thought it might be good to take a snapshot of what's "in" today. I like to think of web development phases starting from idea to delivery... all of it backed by strong principles of how to build great apps.

The following are my core web development principles today:

- Designing for mobile first (even if you're not building a mobile app)
- Build only single page apps
- Create and use your own REST API
- "Sex sells" applies to web apps
If you call yourself a web developer, but don't know JavaScript or CSS, it's time to do some learning.
“Java remains – in spite of the fragmented programming language landscape – a viable, growing language.”

http://redmonk.com/sogrady/2012/02/08/language-rankings-2-2012/
... embraces JavaScript

... is learning mobile frameworks
  - jQuery Mobile, Sencha Touch, PhoneGap or Native

... is using HTML5 and CSS3

... is developing REST APIs with the stateless framework that best supports their language

IE6 is dead, IE7 isn’t far behind...
You have to decide...

Are you a web developer?
You have to decide...

Or are you a services developer?
Inversionism

Client-Side MVC frameworks compared

Published: 13 Feb 2012

A month ago Gordon L. Hempton wrote about twelve JavaScript frameworks in the Client-Side MVC space. His rating criteria were different to mine. One that really sticks out is that I like the logic not forcing the template HTML to migrate to \texttt{<script> tags}. Depending on the sophistication of the app, I like to be able to see the app in a browser or DreamWeaver when the framework is not running. It gives me a way of gauging the composition of the app. It appeals to a WYSIWYG leaning that I have. I like my UI frameworks to be \textit{built for designability} if you like.

Addy Osmani has a number of implementation of a TODO app on a github pages site. For composition purposes, this really is the definitive place presently. Using those, I'm going to scrutinize the HTML and how it the app looks without JavaScript. I checked out Addy's repo, then recursively deleted all javascript files, before loading the main page for each into a browser.
Developer Productivity

What keeps you from doing your work?

- Too much multitasking: 53%
- Boring tasks: 39%
- Bad management: 30%
- Buggy software: 26%
- Lack of motivation: 26%

http://zeroturnaround.com/blog/developer-productivity-report-part-3-developer-efficiency/
There is no “best” framework

Just lots of awesome choices...
Don’t listen to me!

THINK FOR YOURSELF!

MAKE ME!
Choose your own!

- Prioritize a list of features that are important to your application.
- Pick 3-4 frameworks and do a 1-week spike with each, developing the same application.
- Document and rank each framework against your list of features.
- Calculate and choose!
- ... Or just pick one and get to work...
But don’t forget...

Modern Principles in Web Development

By Rich Manalang, Developer Advocate
About Developer
On January 18, 2012

I've been kickstarting a bunch of small web apps lately. It seems like every time I start a new project, there's always something new that causes me to adjust my development principles. I thought it might be good to take a snapshot of what's "in" today. I like to think of web development phases starting from idea to delivery... all of it backed by strong principles of how to build great apps.

The following are my core web development principles today:

- Designing for mobile first (even if you’re not building a mobile app)
- Build only single page apps
- Create and use your own REST API
- "Sex sells" applies to web apps
Questions?

- **Contact**
  - [http://raibledesigns.com](http://raibledesigns.com)
  - @mraible

- **Download**
  - [http://slideshare.net/mraible](http://slideshare.net/mraible)