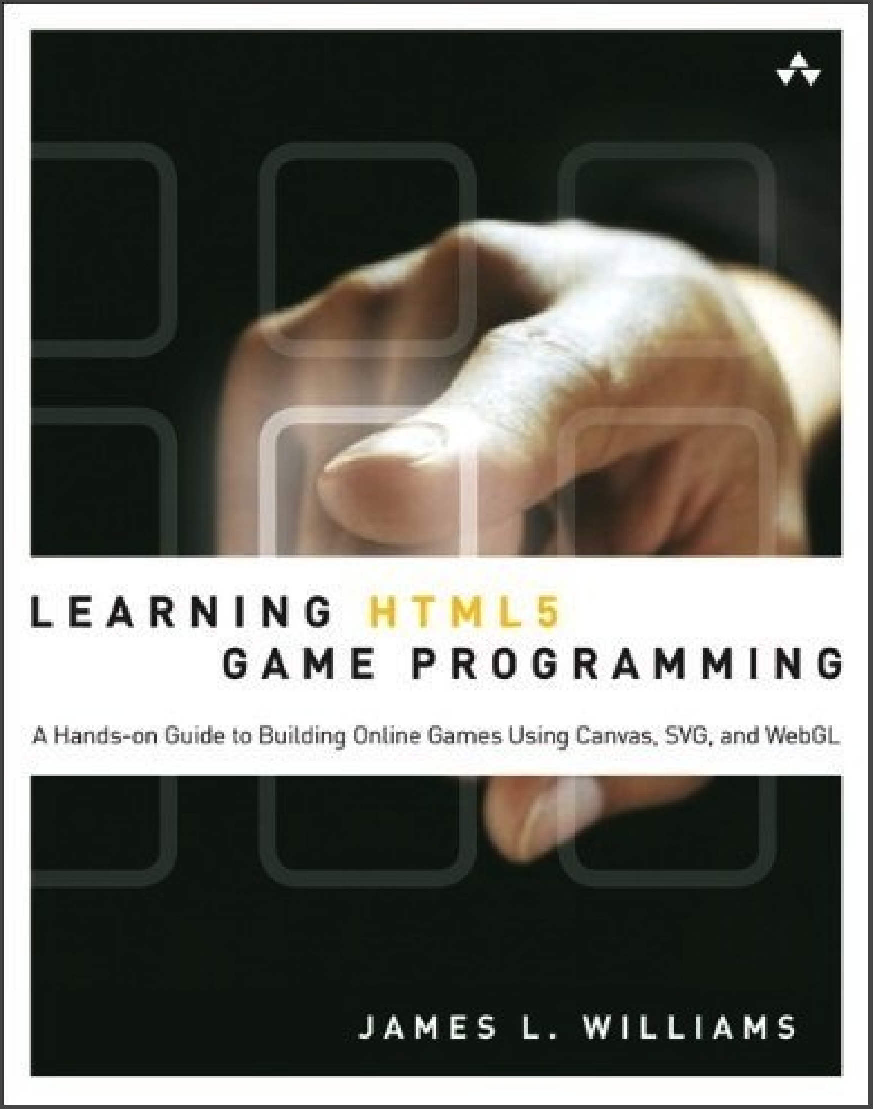
# Ratpack

## Classy and Compact Groovy Web Apps

James Williams

# About Me

- Co-founder of Griffon
- Author of "Learning HTML5 Game Programming" http://amzn.to/HTML5-Game-Book
- Blog: http://jameswilliams.be
- Twitter: @ecspike
- Google+: http://gplus.to/ecspike


LEARNING HTML5
GAME PROGRAMMING
A Hands-on Guide to Building Online Games Using Canvas, SVG, and WebGL
JAMES L. WILLIAMS

# Agenda

- What are Groovy and Ratpack?
- Routes
- Templates
- Running the App
- Deploying to Servlet Containers
- Deployment
- Demos

# What is Groovy?

- Dynamic language on the JVM inspired by Java, Ruby, and Python
- Compiles Java source - yay!
- Can mix Java and Groovy classes and libraries
- Do alot more with less code
- http://groovy.codehaus.org

# What is Ratpack?

- Apache 2 Licensed
- Inspired by Sinatra (Ruby)
- Logic is largely driven by Routes
- Powered by Jetty
- Built-in support for Groovy SimpleTemplateEngine
- Github: https://github.com/bleedingwolf/Ratpack

# Ratpack vs (G)rails

# When to use Grails...

- Data-driven model
- Large number of models with differing endpoints
- Spring Security is a definite need
- Out of the box persistence integration is required
- Plan to make great use of the Grails plugin ecosystem

# When to use Ratpack...

- Low number of domain classes
- Limited number of routes/endpoints
- Model - View - Controller paradigm is not needed
- Small but evolving REST API

# Routes

- Composed of a HTTP verb, an endpoint, and a code block
- Verb can be one of the nine canonical verbs or you can create your own
- Endpoint corresponds to URL fragment.
- :<identifier> injects that value into the urlparams object
- Code block is injected with:
  - request
  - response
  - renderer
  - urlparams

# Routes

```
get("/index") {
    // code
}

post("/login") {
    // code
}

put("/post/:id") {
    // code
}
```

# Routes

```
get("/index") {
    // code
}

post("/login") {
    // code
}

put("/post/:id") { ==>     /post/3
    // code                    urlparams.id == 3
}
```

# Sample Ratpack App

```
import com.bleedingwolf.ratpack.*

class App {
public static void main(String []args) {
def app = Ratpack.app {
set 'public', 'public'
set 'templateRoot', 'templates'
post("/") {
request.toString() + "\n" +params.toString()
}
get("/") {
"Hello world"
}
}
RatpackServlet.serve(app)
}
}
```

# Demo

# Templates

# Templates

- Uses Groovy SimpleTemplateEngine by default
- Can swap in other template engines
- SimpleTemplateEngine can be used in conjunction with others

# SimpleTemplateEngine

- Means to generate text dynamically
- Uses JSP-style instructions for code blocks
  <% %>  and <%= %>
- Can also use Groovy expression evaluators
  ${expression}

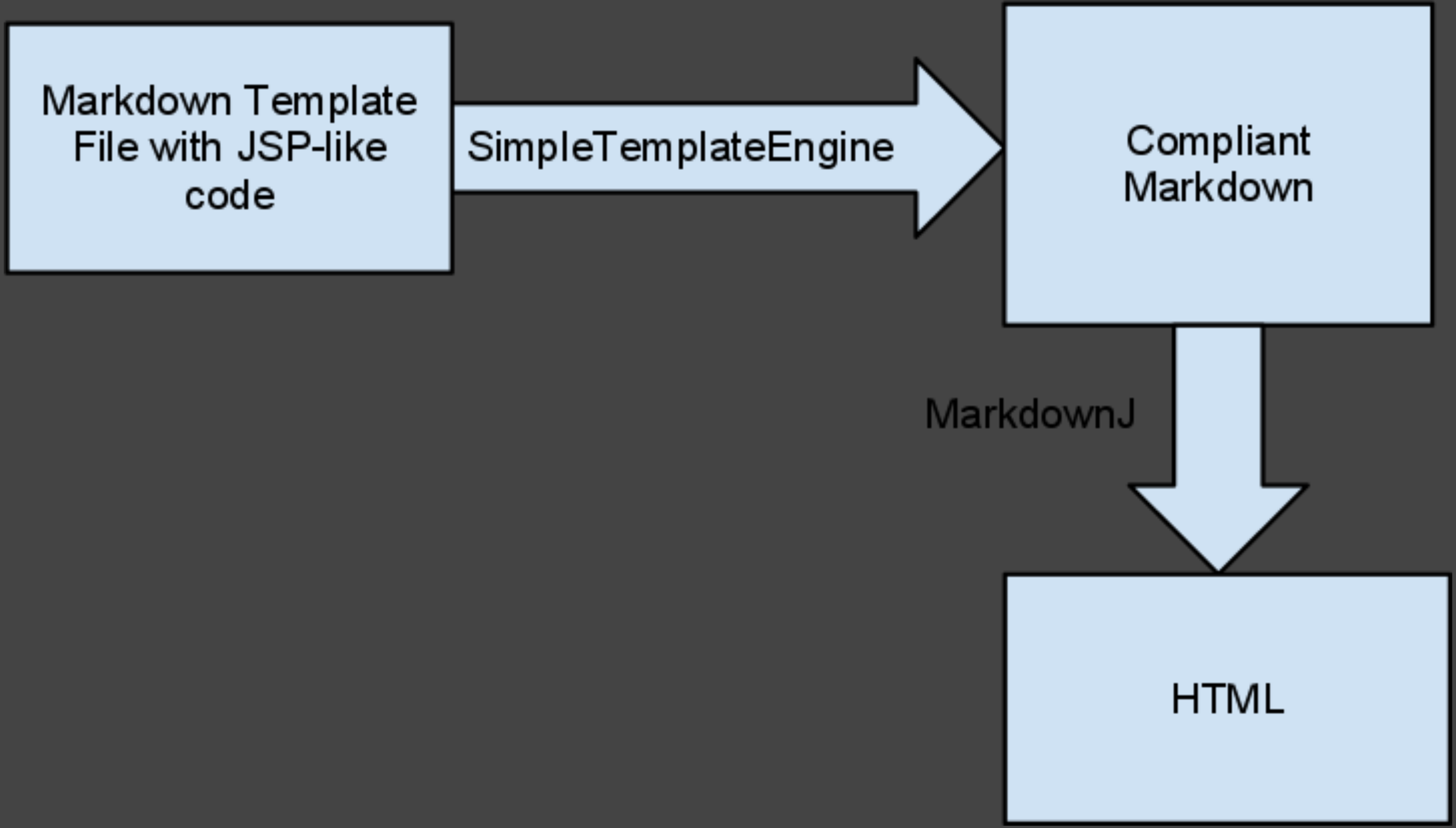
# Markdown

- Created by John Gruber and Aaron Swartz
- Markup language for converting text to (X)HTML
- Supported by many frameworks and websites
- In many different languages
- http://daringfireball.net/projects/markdown/

# Markdown Basics

- Headers (#)
- Lists (* or numbers)
- Code Snippets (`)
- Emphasis (*)
- Strong/Bold (**)
- Links

# Markdown + SimpleTemplateEngine



Markdown Template File with JSP-like code

SimpleTemplateEngine

Compliant Markdown

MarkdownJ

HTML

# Sample Ratpack App

```
import com.petebevin.markdown.*
import com.bleedingwolf.ratpack.*

class App {
public static void main(String []args) {
def app = Ratpack.app {
set 'public', 'public'
set 'templateRoot', 'templates'

                              get("/time") {
render "index.html", [time:new Date()]
}
                           get("/") {
def m = new MarkdownProcessor()
def p = render 'template.md', [num:4]
m.markdown(p)
}
}
RatpackServlet.serve(app)
}
}
```

# Demo

# Creating a Blog App with Ratpack

# App Essentials

- Simple blog engine
- Uses NoSQL for persistence
- Uses a tweaked Tumblr template
- ~ 156 lines of Groovy code
- Source: http://github.com/jwill/Conference-Demos/user-groups

# Blog CRUD Routes

```
get("/entry/create") {
render '/entry/create.html', [contextPath:request.getContextPath()]
}

get("/entry/show/:id") {
def entry = derby.get(urlparams.id)

     render '/entry/show.html', [entry: entry,
        contextPath:request.getContextPath()]
}
get("/entry/delete/:id") {
derby.remove(urlparams.id)
}
```

# Blog CRUD Routes (cont'd)

```
get("/post/list") {
def list = derby.all()
list.sort(sortClosure.curry('dateCreated'))
list = list.reverse()
render '/post/index.html', [posts: list]
}

post("/post/save") {
def post = new Post(params)
derby.save(post.properties, {obj ->
println "Finished saving."
new JSONObject([success:true]).toString()
})
}
```

# Demo

# Running/Deploying the App

# Gradle

- Language independent build system leveraging Groovy
- Build system for Grails, Spock, and many others
- Uses Maven for dependency management
- Vibrant plugin community
- http://gradle.org

# Gradle Application Plugin

- Requires Gradle 1.0+
- Code: plugin 'application'
- Provides common tasks for the application lifecycle.
- Provides the following tasks:
  - run
  - installApp
  - distZip

# Gradle File

```
apply plugin: 'groovy'
apply plugin: 'application'

repositories {
flatDir name:'lib', dirs:'lib'
mavenCentral()
}

dependencies {
groovy group:'org.codehaus.groovy', name:'groovy', version:'1.8.0'
groovy group:'com.bleedingwolf.ratpack', name: 'Ratpack', version:'0.2-SNAPSHOT'
// Ratpack's other dependencies
}
```

# Gradle File (cont'd)

```
installApp {
into('build/install/'+applicationName){
from ('templates').into '/templates'
}
into('build/install/'+applicationName){
from ('public').into '/public'
}
}
distZip {
into(applicationName){
from ('templates').into applicationName+'/templates'
}
into(applicationName){
from ('public').into applicationName+'/public'
}
}

mainClassName = "BlogApp"
```

# App Structure

/ main/src/groovy
/ templates
/ public
/ lib ==> contains Ratpack.jar
/ resources/web.xml ==> only needed for packaging war
/ build.gradle ==> contains Ratpack's maven depos

# Deploying App as a War

# Gradle File Additions

```
war {
into('/public') {
from('public')
}
into('/templates') {
from('templates')
}
classpath fileTree('lib')
webXml = file('resources/web.xml')
}
```

# Web.xml File

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
    PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
<servlet>
        <servlet-name>AppServlet</servlet-name>
        <servlet-class>AppServlet</servlet-class>
</servlet>
<servlet-mapping>
        <servlet-name>AppServlet</servlet-name>
        <url-pattern>/*</url-pattern>
</servlet-mapping>
</web-app>
```

# Questions ?

# Links

MarkdownJ - http://markdownj.org

Ratpack - http://github.com/bleedingwolf/Ratpack

Gradle - http://gradle.org

Demos - http://github.com/jwill/ConferenceDemos/strangeloop