GETTING GRAILS INTO THE ENTERPRISE

HOW TO TALK YOUR FAVORITE TECHNOLOGY INTO YOUR COMPANY AND KEEP IT THERE

GR8CONF.US - JULY 28, 2014

Created by Ryan Vanderwerf / @RyanVanderwerf

ABOUT ME

Chief Architect / Director of Products @ ReachForce
Co-Chair Austin Groovy and Grails User Group
Effective Gradle Implementation Video Screencast Series released soon!
Help maintain Grails Quartz plugin
Maintain GVPS Plugin (Grails Video Pseudo Streamer)
Maintain Struts-1 Plugin
Submit pull requests for others when I can!

WHAT WE WILL COVER

- How to Get Grails In The Door
- My Story With Grails
- Tips to help create a plan to start a project or introduce Grails in your company
- Tips to convince management or project managers
- Tips for selecting a candidate application
- Deploying a plugin to a maven repository / Artifactory
- A little tour of Jenkins
- Plugins to help keep things running smoothly
- Plugins to help you get started and get Grails 'in the door' such as using an old Struts 1.x application


The Magician - 1

HOW TO GET GRAILS (AND GROOVY!) IN THE DOOR

Looking to introduce Grails into your Company or Enterprise?
Make sure you can answer these questions:

Do you have a plan? (Please work with one or it will hurt)i.e. why?
How will you sell it to management? (What's the upside? Risk?)
How will you know it is effective? (Have a way to benchmark 'success')
How will you keep things running smoothly? (Call on your trusty plugins!)
What is a good kind of low-risk project to migrate? (This varies, I'll show a struts example)

MY STORY WITH GRAILS

Old employer had very antiquated Struts/OJB application
Because of technical debt work orders from customers were taking too long to do to be profitable
Wanted a fresh start, but could not afford to re-write all old code
Need a framework that allows for rapid development
Need a solution that Java developers can quickly grasp and understand (and enjoy!)
Can be deployed to existing infrastructure as a WAR file

BARRIERS YOU MAY ENCOUNTER

- Barriers to entry from non-technical staff or managers:
  - Fear of change
  - Skepticism it can deliver
- A little different way of thinking to leverage Grails strengths like scaffolding
- Patience: People will resist it if it doesn't give immediate magical results, so don't promise them
- Difficult for project managers to understand defining the domain model up front saves time later (i.e. regenerating scaffolding repeatedly loses saved time)
- Difficult for project managers/managers to understand why putting work into scaffolding templates up front saves time and money going forward (new process)

PICKING A LEGACY APPLICATION

It is important to pick a good legacy app candidate for introduction

- Greenfield projects are always easier but can be risky if you are new to the framework – Start with some kind of internal tool
- Old Struts 1.x projects can be blended easily without a lot of risk
- Easy win because people will have low expectations on such an old application
- Write unit tests for existing code in Groovy to 'wet' developer appetites (and practice)
- Switch a build to Gradle, start small!

CREATING A PLAN

You must have a plan before you begin - responsible thing to do

- How you will pick an app? old or new?
- cost - how much will it cost if you need outside help?
- who - how many resources do you need? Any IT changes necessary?
- Are you superiors and peers sold on it yet? Did you do a POC?
- It's about business value, this is the language your management will speak. That may be the only information that actually comprehend - compute some basic numbers to help

SELLING TO MANAGEMENT

You must have a plan before you begin - responsible thing to do

- Gain development efficiencies, no constant server restarts during development
- Plugin ecosystem – won't have to re-invent the wheel so many great useful plugins to save time (and most of the source, you can contribute most of them to make them better)
- Easy for Java developers to understand (See 'Making Java Groovy by Ken Kousen')
- Still deploys as a WAR file to the container, no significant infrastructure changes
- Can build a hybrid product to bridge the gap between new a old, giving a long term plan to modernize as you go
- Great community support

FACILITIES YOU WILL NEED TO PUT IN PLACE

You'll need some of these infrastructure items

- CI Server – Hudson/Jenkins most popular free choice, works great with Grails
  - setup jobs on CI Server to run tests - also add code analysis like codenarc/gmetric
  - trigger builds by watching SCM for changes
  - use coverage tools like Cobertura or Emma
  - use code analysis tools like FindBugs or CodeNarc
  - Make your CI server fun, little things like the chuck norris plugin or integration game keep developers entergized and interested (A little competition for a prize never hurts!)
  - Use a service like Cloudbees if you can't run it yourself
- Some kind of deployment tool. Roll your own in Grails, or use artifact deployers in Hudson, or tools like Cargo, Chef, Puppet, Ansible, Docker to help you

STRUTS 1 PLUGIN

You'd be shocked how many big places still heavily use Struts 1!

- Lets you merge Struts applications into a Grails application
- Often overlooked option that works quite well to get Grails 'in the door' to help aging applications
- You can even write Struts actions in Groovy
- Easy win to run old legacy code and new Grails functionality in parallel
- If you still use Grails 1.3.x use plugin v1.3.10. For Grails 2.x, use 1.3.11+

STRUTS 1 PLUGIN

- Handle exceptions well for legacy code. In Grails 1.3.x, exceptions on a 500 server error page in your legacy application will be wrapped in a Grails stack. This will frequently cause blame on Grails for legacy bugs and give opponents ammunition to not go forward with Grails. Stack traces in Grails 2 are much cleaner and also avoid this error

```
multipartResolver(org.codehaus.grails.struts.StrutsAwareMultipartResolver)
 {
    strutsActionExtension = ".do"

}
```

- (If using Grails 2.x plugin version 1.3.11, the plugin does this for you)
- If Using Grails 2, the ControllerActionProxy (TODO to fix this)

STRUTS 1 PLUGIN

Demo

FLEX / BLAZEDS PLUGIN

Another one I've run into before.. you can use this plugin Burt Beckwith wrote a while back

- Lets you blend Grails, Struts, and Spring Security
- I've run into quite a few of these 'quick' apps that were quick fixes that won't die
- Anything to externalize properties better, often I've seen people hard-code into flash files properties (yuk!)
- Has not been updated in a few years, may have issues with Grails 2.3+

FLEX / BLAZEDS PLUGIN

Another one I've run into before.. you can use this plugin Burt Beckwith wrote a while back

- Lets you blend Grails, Struts, and Spring Security
- I've run into quite a few of these 'quick' apps that were quick fixes that won't die - could be another easy vector in
- Anything to externalize properties better, often I've seen people hard-code into flash files properties (yuk!)
- Has a nice user guide and docs thanks to Burt
- Has not been updated in a few years, may have issues with Grails 2.3+

GOOGLE GWT

Another good candidate - may not be best for super complex projects

- Let's you combine a GWT project with Grails
- I have a GWT legacy app, this would be a good candidate if you have one to get Grails in the door
- 1.0 status, less than 6 months since last update
- If you can find a reason to ditch that horrific GWT maven script for compilation, that alone is a win
- Has several maintainers including Peter Ledbrook :) and a good tutorial and documentation
- There is also a SmartGWT plugin but 2 years since last update

CONTINUOUS INTEGRATION SERVER

- Do you have a server to run it on? If yes, you have lots of options:
  - Jenkins / Hudson (recommended for Grails)
  - Cruise Control
  - Continuum
  - Team City
  - Travis
  - Bamboo

CONTINUOUS INTEGRATION SERVER

- If no there are cloud based options:
  - Cloudbees
  - drone.io
  - CircleCi
  - CodeShip
  - Travis
  - Elastic Bamboo
  - Run your own instance of Jenkins/Hudson on a EC2 or other cloud provider OS image (and maintain yourself)

CONTINUOUS INTEGRATION SERVER

Jenkins is by far the most popular now according to Rebellabs

ARTIFACT / MAVEN SERVER

JFrog Artifactory the most popular and easy. Archiva etc also work fine

DATABASE REVERSE ENGINEERING

- 2 Common Options to generate domain model from Database
  - Grails Reverse Engineering Plugin (Only works properly on a separate 1.3.x project due to Hibernate version issues)
  - The GRails Application Generator (GRAG) Standalone application
- Mirror domain objects to legacy tables is key to implementing new features in Grails and leaving legacy code alone (If you won't have hbm files to import to Grails)
- Mirror domain objects to legacy tables is key to implementing new features in Grails and leaving legacy code alone (If you won't have hbm files to import to Grails)
- When you have a domain object and a legacy bean make sure you handle cache consistency when writing objects

DATABASE MANAGEMENT

- Database change management
  - Use Grails Database Migration Plugin (wraps Liquibase)
  - Liquibase Directly
  - Most other plugins like liquibase and autobase are deprecated in favor of the Database Migration Plugin
  - Your DBAs may already have a process and procedure - you may not win this one
- Do NOT use the 'dbCreate' option in Grails for any kind of production system – it is not smart enough to handle field renaming of columns
- If using the Grails Database Migration Plugin, use the changelog.groovy format and not the xml format, due to bugs in the functionality that handles xml changelogs in the plugin
- If you must use the xml format, use Liquibase directly
- Create separate project and install the migration plugin just for it (Due to Hibernate 3 vs 4 versioning issues)

MODULARITY

- Split up major functional areas of the legacy applications into separate plugins, but keep some things in mind:
  - JSPs do not serve well from plugins, start with just the java code, then work your way to converting JSPs to GSPs called from the plugins
  - Beware of cyclic dependencies, Grails does not tolerate them (Good design should avoid this, but sometimes it's hard to break of legacy spaghetti code)
  - Do not use the lib folder for jar files, keep dependencies clean....

DEPENDENCY MANAGEMENT

- When migrating legacy app to Grails format, and you use Ant and not Maven, follow these tips:
  - Don't just copy jars into the grails/lib folder, set up dependencies in BuildConfig.groovy as much as possible(If no repo, use Bintray or Artifactory)
  - Run dependency-report to help work out conflicts
  - Do not use the lib folder for jar files, keep dependencies clean....
- If your legacy project is Maven based, Grails 2.1 has excellent Maven support built in via 'grails create-pom' command (POM demo struts-demo21)

MAVEN PROJECT MANAGEMENT

MAVEN PROJECT MANAGEMENT

- Multi-module support (From docs):

```
create-multi-project-build

grails create-app myapp
grails create-plugin plugin-a
grails create-plugin plugin-b
grails create-multi-project-build com.mycompany:parent:1.0-S
NAPSHOT
mvn install
```

RELEASE PLUGIN

- Used to push plugins to maven repository (or your own plugins to the
- Add the following to your BuildConfig.groovy:

```
        grails.project.repos.default = "PluginSnapShots"
            grails.project.repos.PluginSnapShots.url = "http://127.0.0.1:8081/artifac
pshot-local"
        grails.project.repos.PluginSnapShots.type = "maven"
        grails.project.repos.PluginSnapShots.username = "admin"
        grails.project.repos.PluginSnapShots.password = "password"
```

LOCALIZATION PLUGIN

- Store your i18n message bundles in the database
- Can Import legacy i18n property files from most systems
- Provides Caching
- allows changes to labels on the app without a new build
- Store your i18n message bundles in the database
- Lets non-technical users fill in missing labels for you(and a value add to sell when switching to Grails)
- Combine with the filterpane plugin to add searching ability

FILTERPANE PLUGIN

- Great for adding search ability for larger number of legacy reverse engineered domain objects (and rows within those)
- Simple to install and implement: add a new action to your controller and add some parameters to your list view page

CLUSTERING

- Ehcache or new Spring Cache
- Use your servlet containers http session clustering or use Terracotta
- Use Terracotta open source edition for visibility and cache management of level2 and general cache management
- Hazelcast is also a great option for distributed cache

USEFUL RESOURCES

http://grails.org/plugin/filterpane
http://grails.org/plugin/localizations
http://grag.sourceforge.net/http://grails-plugins.github.com/grails-database-migration/
http://grails.org/plugin/struts1
https://github.com/rvanderwerf/grails-struts1
http://terracotta.org/downloads/open-source/catalog
http://grails.org/plugin/release
http://grails.org/plugin/db-reverse-engineer
http://grails-plugins.github.io/grails-flex/


```
//Feed your brain
gr8.technologies.each
    yourBrain << it
}
```