Application Architecture in Groovy

Dan Woods

Who Am I?

Senior Consultant

Object Partners

@danveloper

github.com/danveloper

danielpwoods@gmail.com

Application Architecture

What makes architecting a Groovy application different?

Application Architecture

What is "Application Architecture"?

The Principles of Application Architecture in Groovy

Principles

Make Code "More Readable"

Principles

Be Explicit with Magic

Principles

Favor Convention over Configuration

Make Judicious Use of the Language Constructs

Make Code "More Readable"

- Closures as Final Argument to Method

Make Code "More Readable"

- Map-based Constructors

Make Code "More Readable"

- Getter/Setter Overrides

Make Code "More Readable"

- Make varargs Maps

Make Code "More Readable"

- Interceptors

Make Code "More Readable"

Build Fluent, Human-Readable APIs

Make Code "More Readable"

- Method Chaining

Make Code "More Readable"

- String Literal Method Names

Make Code "More Readable"

- DSLs that “make sense” of complex operations

Make Code "More Readable"

Extending Groovy

Make Code "More Readable"

- MetaClass Enhancements

`groovy.runtime.metaclass.[package].[class]MetaClass`

Make Code "More Readable"

- Concept-Driven Meta-Object Enhancements

Make Code "More Readable"

- Category-based Enhancements

Make Code "More Readable"

Still not enough?

Change the Language!

Make Code "More Readable"

- Global & Local AST transforms

example: http://github.com/danveloper/gdi

Favor Convention over Configuration

Groovy Ecosystem is Convention-Favorable

Be Explicit with Magic

Don't Always Extend with a MetaClass

Be Explicit with Magic

Make It Clear Through Convention Where Magic Comes From

Favor Convention over Configuration

Where do I put things?

Favor Convention over Configuration

Group by Concept

```
+ groovy.runtime.metaclass
   +- com.danveloper
      +- datamodel
         +- CarMetaClass.groovy
      +- locators
         +- MechanicLocatorMetaClass.groovy

+ com.danveloper
   +- datamodel
      - Car.groovy
      - Truck.groovy
   +- services
      +- mechanics
         - CarMechanic.groovy
         - TruckMechanic.groovy
   +- extensions
      - StaticLookupCategory.groovy
      - StaticCreationCategory.groovy
   +- locators
      - MechanicLocator.groovy
```

Favor Convention over Configuration

Think Grails & Duck Typing

the end.