Simplicity *itself*

Grails Plugins

Reusable packages of Grails functionality

Plugins can...

- provide artifacts
  - controllers, domain classes, custom tags, etc.
- provide Spring beans
- configure the application
- customise the web descriptor

Example - Shiro

- Filters
- Custom artifact (realms)
- GSP tags
- Services
- Spring beans
- Commands

Example - Mail

- Service (*mailService*)
- Dynamic methods (*sendMail*)
- JavaMail library dependency

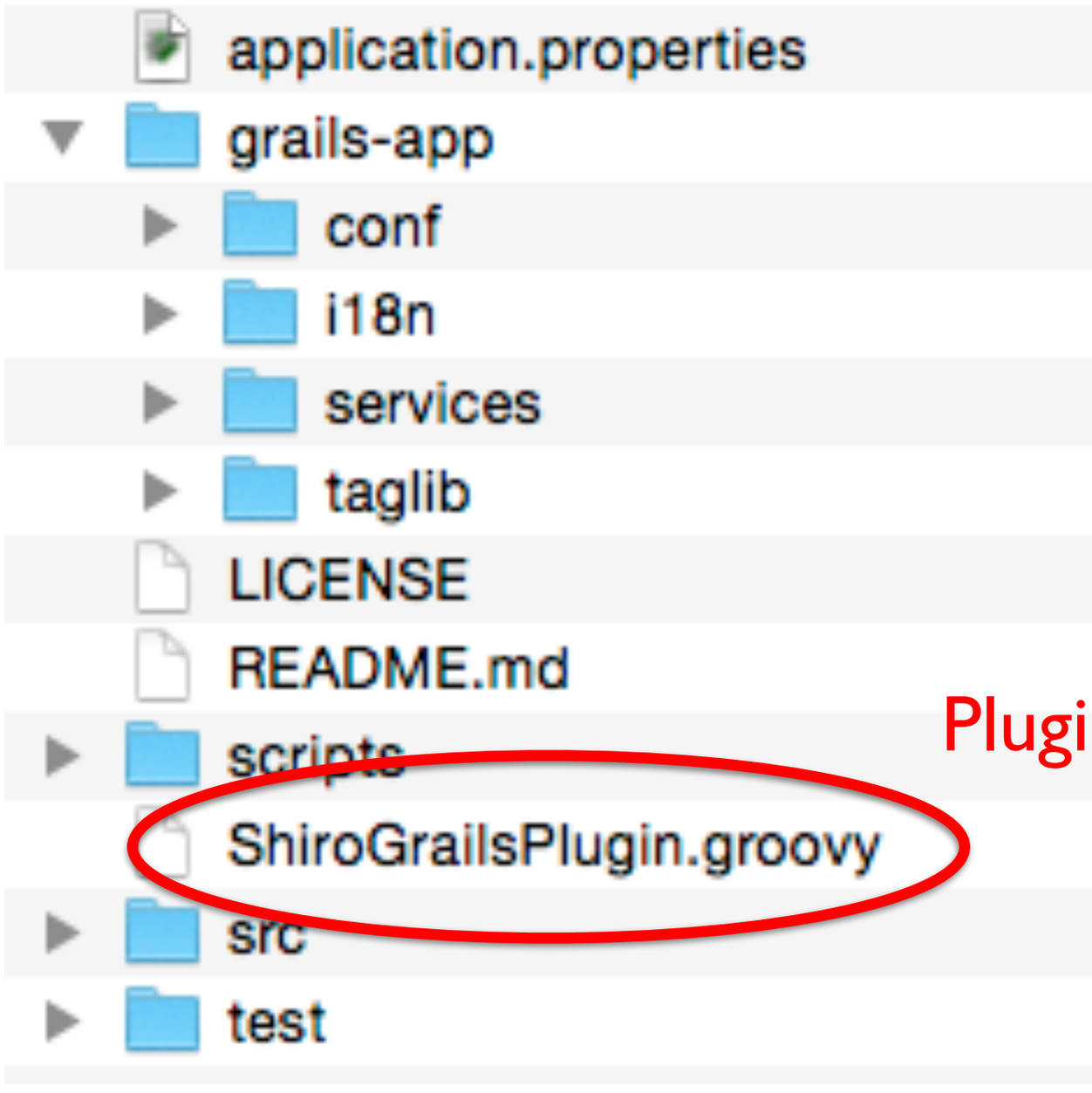
All artifacts provided by a plugin are automatically available to the applications that use the plugin.

For example, plugin domain classes are automatically mapped to the app's database and you can query on them from the app.

Plugin controllers can handle HTTP requests to the app either via convention URL mapping or through custom URL mappings you add to the app.

```
grails create-plugin my-plugin
```

Like an app, but...



Plugin descriptor

```
class ShiroGrailsPlugin {

    def version = "1.2.2-SNAPSHOT"
    def grailsVersion = "1.2 > *"
    def author = "Peter Ledbrook"
    def authorEmail = "peter@cacoethes.co.uk"
    def title = "Apache Shiro Integration for Grails"
    def description = """\
Enables Grails applications to take advantage of the Apache Shiro security layer, adding
"""
    def organization = [name: "nerdErg", url: "http://nerdErg.com/"]
    def developers = [[name: "Peter Ledbrook", email: "peter@cacoethes.co.uk"], [name: "P
    def documentation = "http://grails.org/plugin/shiro"
    def license = "APACHE"
    def issueManagement = [system: "JIRA", url: "http://jira.grails.org/browse/GPSHIRO"]
    def scm = [url: "https://github.com/pledbrook/grails-shiro"]
```

Lots of metadata!

Plugin descriptor hooks

doWithSpring	*Define Spring beans using Bean Builder notation*
doWithApplicationContext	*Work with the Spring application context after it has been initialised*
doWithWebDescriptor	*Add servlet and filter definitions, modify session timeout, etc.*
onChange	*React to reloading events during* `run-app`

What are plugins good for?

- Cross-cutting concerns
  - Access control, email, auditing
- Vertical features
  - Student registration, class scheduling

How do you add plugins to a project?

```
grails.project.dependency.resolution = {
    // inherit Grails' default dependencies
    inherits("global") {
        // specify dependency exclusions here; for exampl
        // excludes 'ehcache'
    }
    plugins {
        // plugins for the build system only
        build ":tomcat:7.0.47"

        // plugins for the compile step
        compile ":scaffolding:2.0.1"
        compile ':cache:1.1.1'

        // plugins needed at runtime but not for cc
        runtime ":hibernate:3.6.10.6" // or ":hiber
        runtime ":database-migration:1.3.8"
        runtime ":jquery:1.10.2.2"
        runtime ":resources:1.2.1"
```

BuildConfig.groovy

Sometimes you will see this in BuildConfig.groovy:

```
grails.plugin.location.'spring-security-cas'="plugins/spring_security_cas.git"
grails.plugin.location.'sghe-zk-core'="plugins/sghe_zk_core.git"
grails.plugin.location.'banner-ui'="plugins/banner_ui.git"
grails.plugin.location.'banner-core'="plugins/banner_core.git"
grails.plugin.location.'i18n-core'="plugins/i18n_core.git"
grails.plugin.location.'banner-packaging' = "plugins/banner_packaging.git"
grails.plugin.location.'banner-aurora' = "plugins/banner_aurora.git"
grails.plugin.location.'banner-general-utility' = "plugins/banner_general_utility.git"
grails.plugin.location.'domain-extension'="plugins/domain_extension.git"
```

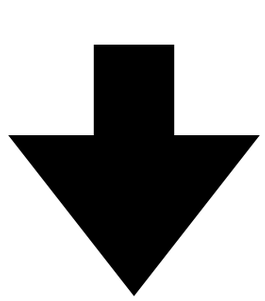
Called in-place or inline plugins. Not recommended!

Publishing plugins

- Use Release plugin
  - Added by default when you create a plugin
- Packages as a source zip
- Deploy to local Maven cache or a Maven-compatible repository

When developing a plugin

```
grails maven-install
```



Add `mavenLocal()` to repository definitions of target app

When developing a plugin

For public plugins:

```
grails publish-plugin
```

For private plugins:

```
grails maven-deploy
```

Remember: everything except the plugin descriptor is in the same relative location as for an app!