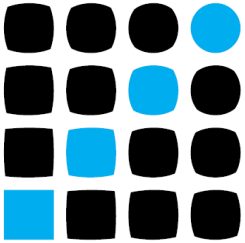


adaptTo()

APACHE SLING & FRIENDS TECH MEETUP
BERLIN, 26-28 SEPTEMBER 2016

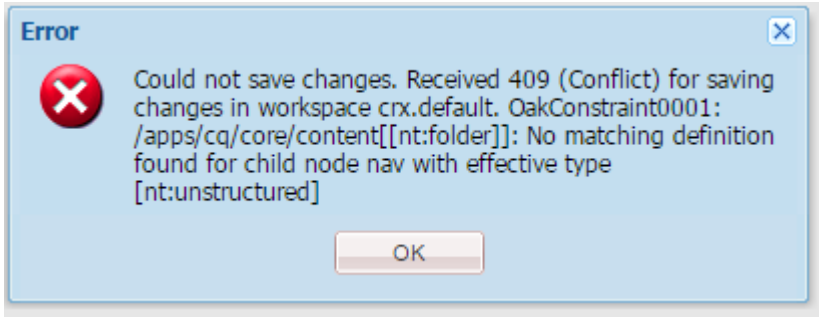
5000+ unattended AEM installations

...in the last four years - is it DevOps?

Georg Henzler - Netcentric

adaptTo()

```
14.09.2016 21:26:17.939 *ERROR* [0:0:0:0:0:0:0:1 [1473881177938]
GET /content/geometrixx/en/products.html HTTP/1.1]
org.apache.sling.servlets.get.impl.DefaultGetServlet No renderer for
extension html, cannot render resource JcrNodeResource,
type=geometrixx/components/widecontentpage, superType=null, path=/
content/geometrixx/en/products/jcr:content
```



5000+ unattended installations… **AND MANY FAILED ONES!**

HTTP ERROR: 503

Problem accessing /system/health. Reason:

`AuthenticationSupport service missing. Cannot authenticate request.`

HTTP ERROR: 503

Problem accessing /libs/granite/core/content/login.html. Reason:

`ServletResolver service missing, cannot service requests`

Powered by Jetty://

parallel (() -> {run} , () -> {talk})

{Block 1} – The Target State

The Target State at 15'ooo feet

- A usable CMS that works according functional requirements for editing (authors) and displaying web sites (end users)
  - A Clone (Project Team for evolving/testing)
  - Production (the world, now this is Cont. Delivery?)

The Target State – Tech at 14'ooo feet

- A CMS System with correct
  - Software
  - Configuration
  - Data

The Target State – Tech at 13'ooo feet

- A CMS System with correct
  - Bundles & Scripts (correct version!)
  - OSGi-/Cloud-/Context-Aware Config for Env
  - Content (what content is right?)

The Target State – Tech at 12'ooo feet

- A CMS System with correct
  - Content Package SW
  - Content Package Conf
  - Content Package Content

The Target State – Tech at 11'ooo feet

- A CMS System with correct
  - Content
  - Content
  - Content

The Target State – Tech at 10'ooo feet

Just install "a bunch of content packages" and you are done!

…so what is hard then?

The Target State – Down to earth.

- …with correct Software - Challenges
  - Product base, hotfixes, feature packs, custom code
  - Dependencies: fail fast for bundles (and OSGi Services, SCR), fail late for scripts, fail “depends” for packages, fail often too late “Http Dependencies”,
  - Many different installation options exist
  - Bundles usually loaded asynchronously via JCR (race conditions, Install Hooks)

The Target State – Down to earth.

- …with correct Configuration: Challenges
  - Config of one environment is only testable there
  - New environments need new configuration
  - Configuration via “Overlay”
  - Security Keys
  - Some configurations are required before first system start
  - Duplicated values in OSGi Configs

The Target State – Down to earth.

- …with correct Content: Challenges
  - The Size of Prod Content
  - Disk Space Requirement for Installation: 3x size
  - Demo/Reference/Test Content
  - Different version of the software may require (sightly) different content structures
  - Creation of content from scratch is rarely tested
  - Content must not be Env-Dependent
  - Setup users, roles and permissions

The Target State – Down to earth.

- …with correct Content Packages: Challenges
  - Installation order matters
  - Correct Filter Rules (deleting nodes, etc.)
  - Masters live at different locations (SW/Config in SCM, Content on PROD)
  - Expanded content does not carry a version
  - Stale pauseInstallation nodes

{Block 2} – Handcrafted Install

Minimal Runbook

- Double-click aem-quickstart.jar and go straight to Geometrixx… enjoy!
- For Sling: launchpad.jar

Manual Runbook

- Unzip quickstart
  - Use `java -jar quickstart.jar -unpack`
- Adjust start script in crx-quickstart/bin
  - Runmodes, memory, JAAS, etc.
- Prepare crx-quickstart/install
  - Bootstrap configurations (repository, /etc/key)
  - Service Pack & Hotfixes

Manual Runbook (cont.)

- Start CQ using start script in crx-quickstart/bin
- Install Hotfixes
- Install Software
  - 3rd Party
  - Platform/SW Configuration
  - Site Software
- Content

adaptTo()

{Block 3} – Machine-crafted

Automation Basics

- Sequence of tasks
  - Is-required detection (to potentially skip task)
  - Tasks-done detection (to proceed to next task)
- Failure
  - Detection is important
  - Recovery possible?
  - If no: “The art of meaningful error messages”

Design Goals

- Atomic commands
  - For installation from scratch / delta update
  - Instance Reset
- Ease of Use
  - Server definition files that are self-descriptive and diffable
  - Usable by developers that know nothing about AEM (yet)

Design Goals (cont.)

- Reach Target State as quick as possible
  - Minimize initial setup time
  - Minimize mean time to failure
  - Make it easy/quick to apply instance deltas
- Minimize resource requirements (Disk, Memory)
- Introduce as little “tool lock-in” as possible
  - Use CRX Content Packages instead of custom syntax
- Minimize Prerequisites

Available Tools

- Puppet
- Chef
- Ansible
- Maven
- Hypervisor + Vagrant
- Docker
- JDK
- node
- Ruby
- Python
- Java
- Javascript

...but no ready-made solution available for AEM in any of those.

Base Software - Approaches

- Sling Provisioning Model
- Sling Crankstart
- Sling Launchpad
- Sling Testing Server Setup Tools
- Sling Docker
- **AEM Quickstart** ← the only Adobe supported option

Supplying files

- Package Managers
  - Maven Artifacts, RPMs, RubyGems, npm modules
  - Maven SNAPSHOT semantic is extremely useful
- Package vs. Source
  - Package: Good for binaries, well-versioned
  - Source: Using synced branches can be elegant and simple

Installation Snapshots

- **Pro**
  - Quicker Installation Time
  - Potentially less automation effort (if snapshot is manually created)
- **Con**
  - Large files, one for author and publish needed
  - No incremental update
  - Download time can easily „eat up“ the initial time saving
  - Snapshot does not easily show what is included (not diffable)
  - Snapshots easily become „Snowflake Servers“

Text-based Server Definition

- **Pro**
  - Diffable
  - Branchable
  - Self-describing
  - Small size
  - Incremental update possible
- **Con**
  - Potentially longer startup time
  - More potential to fail (as more tasks are executed)

Startup Done Check: OSGi Start Level

- OSGi Start Level is exposed via JMX (30 -> fully started)
- Local JMX connection does not require additional configuration

```
VirtualMachine vm = VirtualMachine.attach(pid);
final String LOCAL_CONNECTOR_ADDRESS =
"com.sun.management.jmxremote.localConnectorAddress";
vm.loadAgent(vm.getSystemProperties().getProperty("java.home")
    + "/lib/management-agent.jar");
connectorAddress = vm.getAgentProperties().getProperty(LOCAL_CONNECTOR_ADDRESS);
JMXServiceURL jmxServiceUrl = new JMXServiceURL(connectorAddress);
JMXConnector connector = JMXConnectorFactory.connect(jmxServiceUrl);
MBeanServerConnection mbeanConn = connector.getMBeanServerConnection();
// from here, query object name "osgi.core:type=framework,*"
//                  attribute  "FrameworkStartLevel"
```

Ready Checks for Running Instances

- Quiet log file
  - Works well on local instances (non-shared)
- Sling Health Check
  - Sensible set of tags required
- URL Check for Response Codes
  - Can be done via health checks

Automating Package Installation

- Control order
  - Package Manager Dependencies (works for crx-quickstart/install and sub packages)
  - `curl` to package manager in correct order
- Know your properties.xml
  - `subPackageHandling` (e.g. `*;extract`)
  - `acHandling` (e.g. `merge`)
  - `installhook.actool.class`

Automating Package Installation (cont.)

- Install-Folder vs. Package Manager
  - Service Packs, some hotfixes and basic configurations in Install Folder
  - Most Hotfixes and Software via Package Manager
- Filter Rules
  - Prefer include/exclude over mode merge/update
  - Define a master for every content location (SCM or PROD Content)
- Install Hooks

Automating Package Installation (cont.)

- Beware of pause installation nodes
  - at `/system/sling/installer/jcr/pauseInstallation/*`
- To compare if a package is installed
  - Get list of installed packages from /crx/packmgr/list.jsp
  - Use `new PackageManagerImpl().open(packageFile, false)` to retrieve a ZipVaultPackage that exposes package meta data
  - Filename does often not match

Full Stack

- Apache / Varnish Config is part of the application
  - No standard deployment process
  - Solution for environment specific values required
- Full Stack incl. MongoDB, Apache, etc. should be automated

The Buzz

- DevOps: Server definition files can be used for local dev setup, staging instances and production
- Microservices:
  - OSGi is very modular, just not in separate processes
  - A good caching strategy is counterpart to achieve performance today
  - OAK is not optimized for elastic scaling (yet)

talk.join(run)

Links

Sling

https://sling.apache.org/documentation/development/slingstart.html
https://sling.apache.org/documentation/the-sling-engine/the-sling-launchpad.html

Martin Fowler‘s Snowflake Server

http://martinfowler.com/bliki/SnowflakeServer.html

AEM with Puppet

https://forge.puppet.com/bstopp/aem
https://github.com/bstopp/puppet-aem

Questions