

Introduction to Fedora 4 Features

Learning Outcomes

Understand the purpose of a Fedora repository

Understand the core features of the software

What is a Fedora Repository?

Secure software that stores, preserves, and provides access to digital materials

Supports complex semantic relationships between objects inside and outside the repository

Supports millions of objects, both large and small

Capable of interoperating with other applications and services

Exposing and Connecting Content

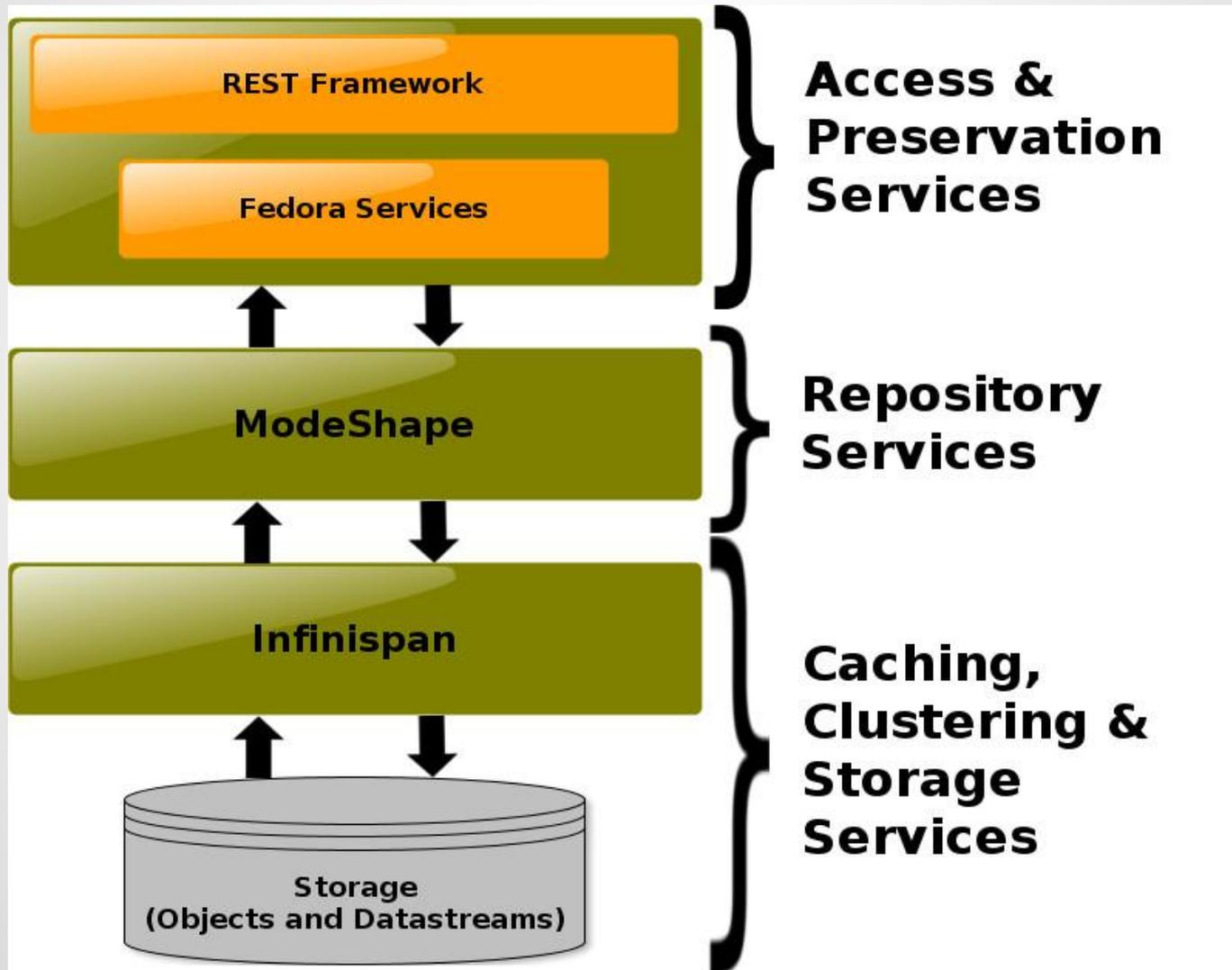
Flexible, extensible content modeling

Atomic resources with semantic connections
using standard ontologies

RDF-based metadata using Linked Data

RESTful API with native RDF response format

Component Stack



Standards

Focus on existing standards

Fewer customizations to maintain

Opportunities to participate in related communities

Core Features

Core Features and Standards

CRUD - *Linked Data Platform (LDP)* ✓

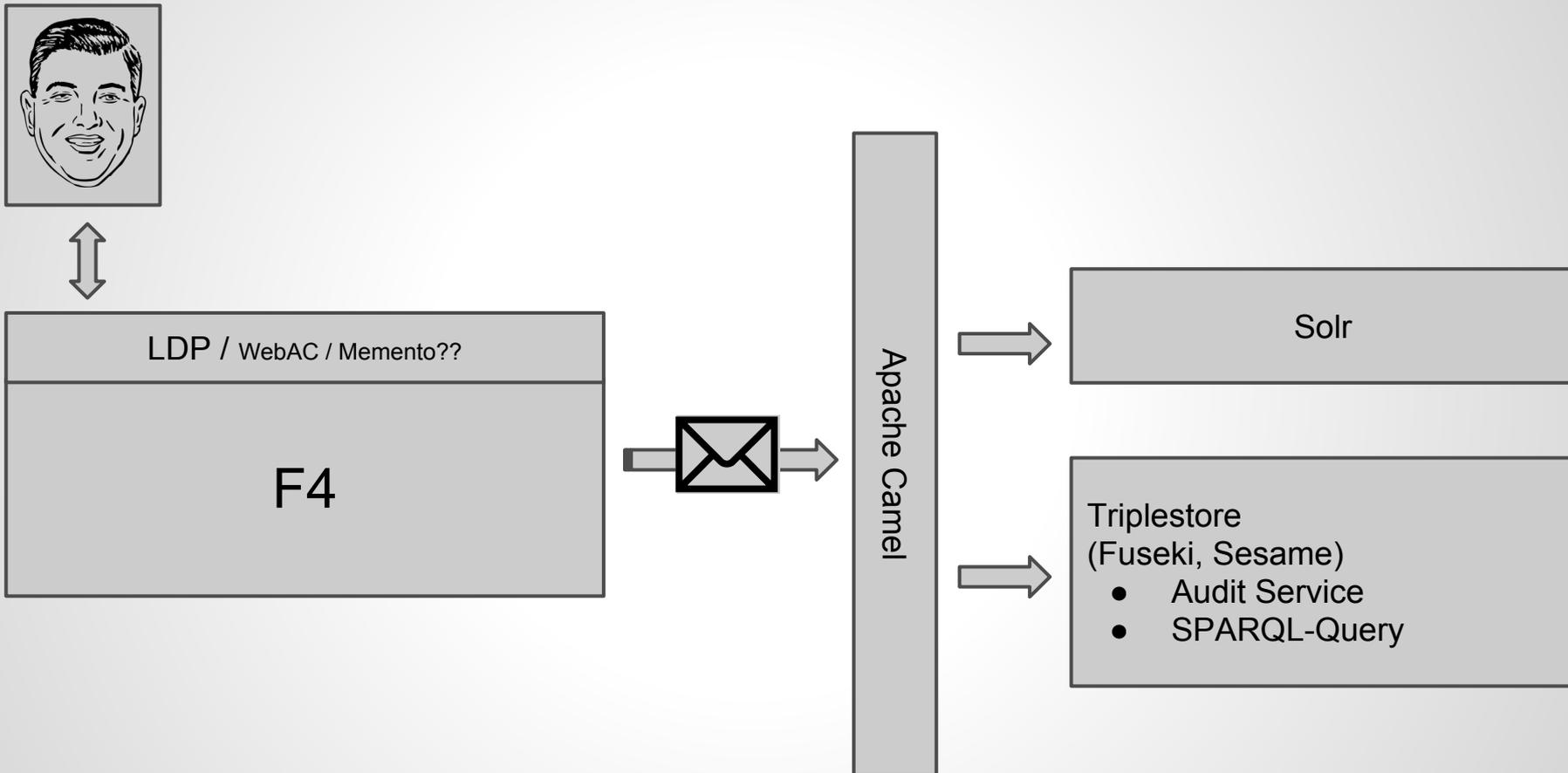
Versioning - *Memento*

Authorization - *WebAC* ✓

Batch Atomic Operations - (*a standard??*)

Fixity - <http://tools.ietf.org/html/rfc3230#section-4.3.2> ✓^{1/2}

Fedora Vagrant Components



Hands-on: CRUD

<http://localhost:8080/fcrepo/rest/>
(fedoraAdmin:secret3)

Create a “cover” Container

PUT vs. POST

...Note: names in demo are only for readability

Make “cover” a pcdm:Object

```
PREFIX pcdm: <http://pcdm.org/models#>
```

```
INSERT {
```

```
  <http://localhost:8080/fcrepo/rest/cover>
```

```
  rdf:type
```

```
  pcdm:Object
```

```
}
```

```
WHERE { }
```

REDUX

Make “cover” a pcdm:Object

PREFIX pcdm: <<http://pcdm.org/models#>>

INSERT { <> a pcdm:Object }

WHERE { }

Batch Atomic Operations (BatchOps)

Multiple actions can be bundled together into a single repository event (BatchOps)

BatchOps can be rolled back or committed

Can be used to maintain consistency

Hands-on: BatchOps



Authorization

The authorization framework provides a plug-in point within the repository that calls out to an optional authorization enforcement module

Currently, four authorization implementations exist:

- No-op
- Role-based
- XACML and
- **WebAC**

Hands-on: AuthZ

Create following Containers

- “files”

...contained inside “cover”

- “my-acls”

...at top-level is fine

- “acl”

...contained inside “my-acls”

- “authorization”

...contained inside “acl”

Final result (structure)

- cover/
 - files/

- my-acls/
 - acl/
 - authorization/

Final result (structure)

“cover” must point to its ACL

- cover/
 - files/
 - my-acls/
 - acl/
 - authorization/
- Diagram: A red arrow originates from the 'files/' sub-item under 'cover/' and points to the 'acl/' sub-item under 'my-acls/'. The text 'acl:accessControl' is positioned near the arrow's path.

- *An ACL must have one or more authorizations*
- *“authorizations” define:*
 - *agent(s)*
 - *mode(s)*
 - *resource(s) or class*

Define the “authorization”

PREFIX acl: <http://www.w3.org/ns/auth/acl#>

PREFIX pcdm: <http://pcdm.org/models#>

INSERT {

<> a acl:Authorization ;

acl:accessToClass pcdm:Object ;

acl:mode acl:Read, acl:Write;

acl:agent "adminuser" .

} WHERE { }

Link “acl” to “cover”

-- Update “cover” resource --

PREFIX acl: <http://www.w3.org/ns/auth/acl#>

INSERT {

 <> acl:accessControl </fcrepo/rest/my-acls/acl>

} WHERE { }

Verify AuthZ

**** *Warning cURL sighting* ****

```
curl -i http://localhost:8080/fcrepo/rest/cover  
> 401
```

```
curl -i -ufedoraAdmin:secret3 http://localhost:8080/fcrepo/rest/cover  
> 200
```

```
curl -i -uadminuser:password2 http://localhost:8080/fcrepo/rest/cover  
> 200
```

```
curl -i -utestuser:password1 http://localhost:8080/fcrepo/rest/cover  
> 403
```

Versioning

Versions can be created on resources with an API call

A previous version can be restored via the REST-API

Hands-on: Versioning

Create version “v0” of “cover”

*** Warning cURL sighting ***

```
curl -ufedoraAdmin:secret3 -i -XPOST -H"slug: v0"  
localhost:8080/fcrepo/rest/cover/fcr:versions
```

Add dc:publisher to “cover”

```
INSERT {
```

```
  <> dc:publisher "The Press"
```

```
}
```

```
WHERE { }
```

Create version “v1” of “cover”

```
curl -ufedoraAdmin:secret3 -i -XPOST -H"slug: v1"  
localhost:8080/fcrepo/rest/cover/fcr:versions
```

* Inspect and Revert

Hands-on: Fixity

Fixity

Over time, digital objects can become corrupt

Fixity checks help preserve digital objects by verifying their integrity

On ingest, Fedora can verify a user-provided checksum against the calculated value

A checksum can be recalculated and compared at any time via a REST-API request

Create some cover binaries

...contained inside “files”

cover.jpg

cover.tif

* Fixity

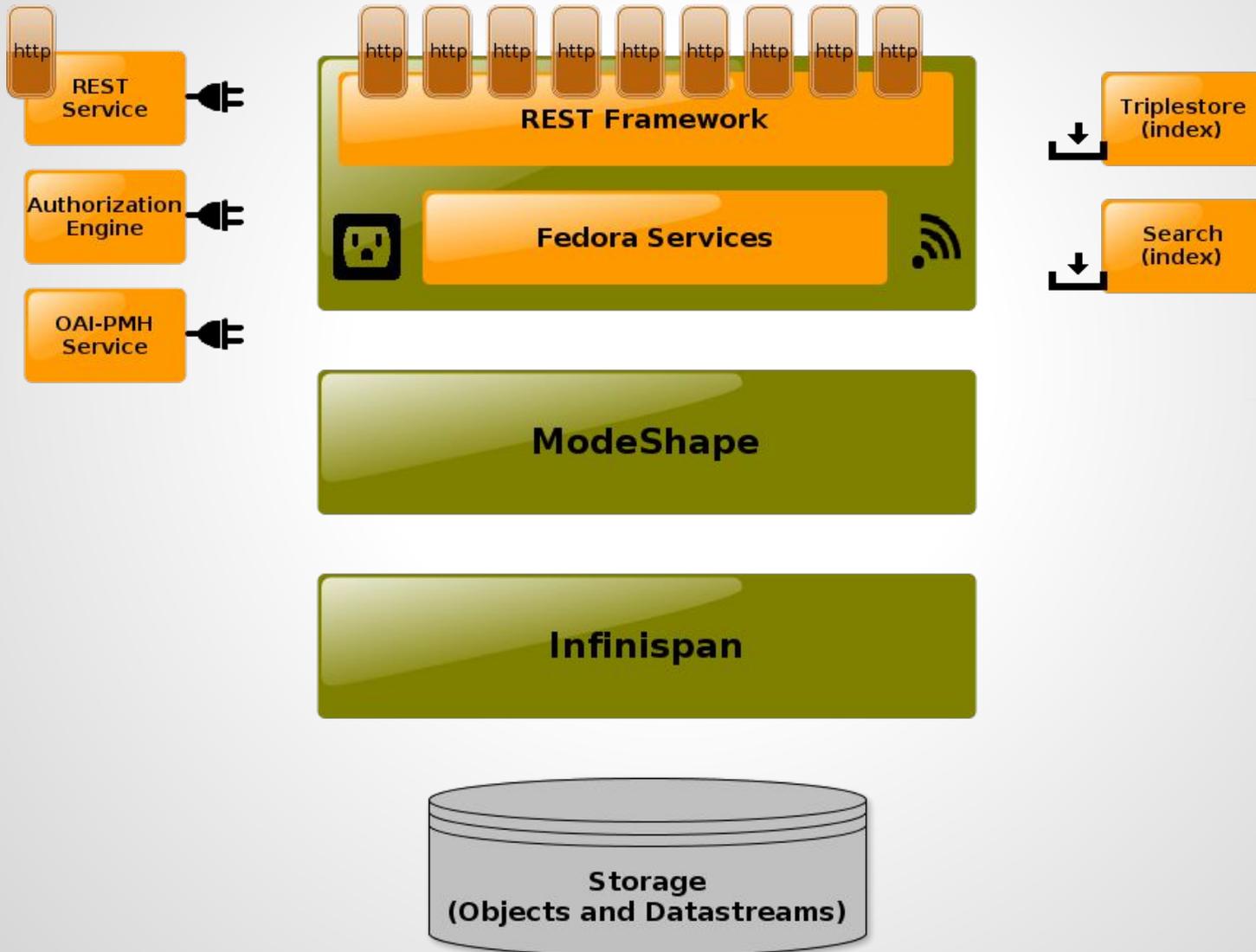
* Corrupt and test?

Non-core Features

Two Non-Core Feature Types

1. External components
 - Consume and act off repository messages
2. Optional, pluggable components
 - Separate projects that can interact with Fedora 4 using a common pattern

Component Architecture



External Component Integrations

Leverages the well-supported Apache Camel project

- Camel is middleware for integration with external systems
- Can handle any asynchronous, event-driven workflow

External - Indexing

Index repository content for search

Content can be assigned the `rdf:type` property "Indexable" to filter from non-indexable content

Solr has been tested

External - Triplestore

An external triplestore can be used to index the RDF triples of Fedora resources

Any triplestore that supports SPARQL-update can be plugged in

Fuseki, Sesame, BlazeGraph have been tested

External & Pluggable - Audit Service

Maintains a history of events for each repository resource

Both internal repository events and events from external sources can be recorded

Uses the existing event system and an external triplestore

Pluggable - OAI Provider

fcrepo4-oaiprovider implements Open Archives Protocol Version 2.0 using Fedora 4 as the backend

Exposes an endpoint which accepts OAI conforming HTTP requests

Supports oai_dc out of the box, but users are able to add their own metadata format definitions to oai.xml

Pluggable - SWORD Server

SWORD is a lightweight protocol for depositing content from one location to another

fcrepo4-swordserver implements 2.0 AtomPub Profile, using Fedora 4 as the backend

SWORD v2 includes AtomPub CRUD operations

Success!

F4: External Integrations

Introducing Camel

What is Camel?

Good question. See: <http://camel.apache.org/what-is-camel.html>

Too many buzzwords - what exactly is Camel?

Okay, so the description above is technology focused. There's a great discussion about Camel at **Stack Overflow**.

So really you want see this: <http://stackoverflow.com/questions/8845186/what-exactly-is-apache-camel>

In short...

- Camel is a framework for creating small message based applications... and then some.
- Camel formalizes working with messages so well it can be described in multiple formats: Java, Spring/Blueprint XML, and Scala.
- Camel is all the code you should not have to write in order to work with queues, files, databases, RESTful APIs, common data formats, command line utilities, etc... in a consistent and reliable manner.

Available Camel Components

<http://camel.apache.org/components.html>

- ActiveMQ
- AWS SQS
- DropBox
- System calls
- Local files
- FTP
- HTTP resources
- LDAP
- SMTP
- SQL
- Twitter
- etc, etc, etc

Camel can run...

- As a stand-alone Java application
- In a servlet container like Tomcat or Jetty
- In an OSGi runtime such as Karaf

What is OSGi?

- Open Service Gateway Initiative
- Framework for modularizing and deploying Java applications
 - Hot deployment
 - Automatic reloading of configuration
 - Sophisticated dependency resolution
 - XML scripting for complex deployments (features)

Hot Deployment

Bundles can be started, stopped, updated, etc... at runtime!

In other words:

**YOU DO NOT HAVE TO RESTART
YOUR SERVER TO UPDATE CODE OR
CONFIGURATION**

Terminology

- Apache Camel
 - Endpoints
 - Components
 - Messages
 - Routes
- Apache Karaf -- OSGi
 - Bundles
 - Features

Hands-On: Into the Vagrant

> vagrant ssh

or:

> ssh -p 2222 vagrant@localhost

password = vagrant

Detour: Fixity corruption revisited

- > cd /var/lib/tomcat7/fcrepo4-data/fcrepo.binary.directory
- > sudo su
- > find . -name *[cover.jpg-sha1]*
- > echo hello >> *[full-path-from-previous-command]*
- > exit

Hands-On: Inspect features

```
> /opt/karaf/bin/client
```

```
>> feature:list | grep fcrepo
```

```
fcrepo-camel
```

```
fcrepo-indexing-triplestore
```

```
fcrepo-audit-triplestore
```

```
fcrepo-indexing-solr
```

```
fcrepo-reindexing
```

```
fcrepo-fixity
```

Hands-On: Helpful Commands

```
>> feature:install fcrepo-audit-triplestore
```

```
>> feature:stop <whichever>
```

```
>> camel:route-list
```

```
>> bundle:list | grep fcrepo
```

```
>> ctrl-d
```

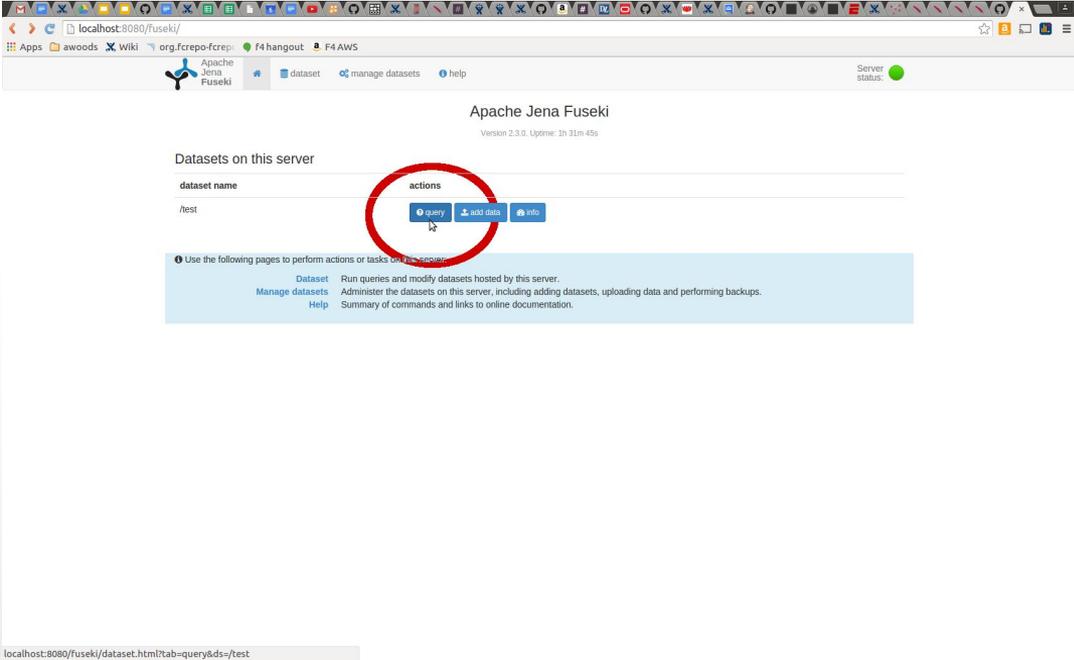
Hands-On: Watch the log

In a new vagrant ssh terminal:

```
> sudo tail -f /opt/karaf/data/log/karaf.log
```

Hands-On: Indexing in triplestore

http://localhost:8080/fuseki



The screenshot shows the Apache Jena Fuseki web interface. The browser address bar displays 'localhost:8080/fuseki/'. The page title is 'Apache Jena Fuseki' with the version '2.3.0' and 'Uptime: 1h 31m 45s'. The main content area is titled 'Datasets on this server' and contains a table with one dataset named '/test'. The 'actions' column for this dataset has three buttons: 'query', 'upload data', and 'info'. The 'query' button is circled in red. Below the table, there is a light blue box with instructions: 'Use the following pages to perform actions or tasks on the server:' followed by a list of links: 'Dataset' (Run queries and modify datasets hosted by this server.), 'Manage datasets' (Administer the datasets on this server, including adding datasets, uploading data and performing backups.), and 'Help' (Summary of commands and links to online documentation.).

localhost:8080/fuseki/dataset.html?tab=query&ds=/test

Hands-On: Indexing in triplestore

```
select * where {  
  <http://localhost:8080/fcrepo/rest/cover> ?p ?o  
}
```

Hands-On: Indexing in triplestore

PREFIX ldp: <http://www.w3.org/ns/ldp#>

PREFIX ebucore: <http://www.ebu.ch/metadata/ontologies/ebucore/ebucore#>

```
select * where {  
  ?s ldp:contains ?o .  
  ?o ebucore:hasMimeType ?m  
}
```

Hands-On: Indexing in triplestore

Audit

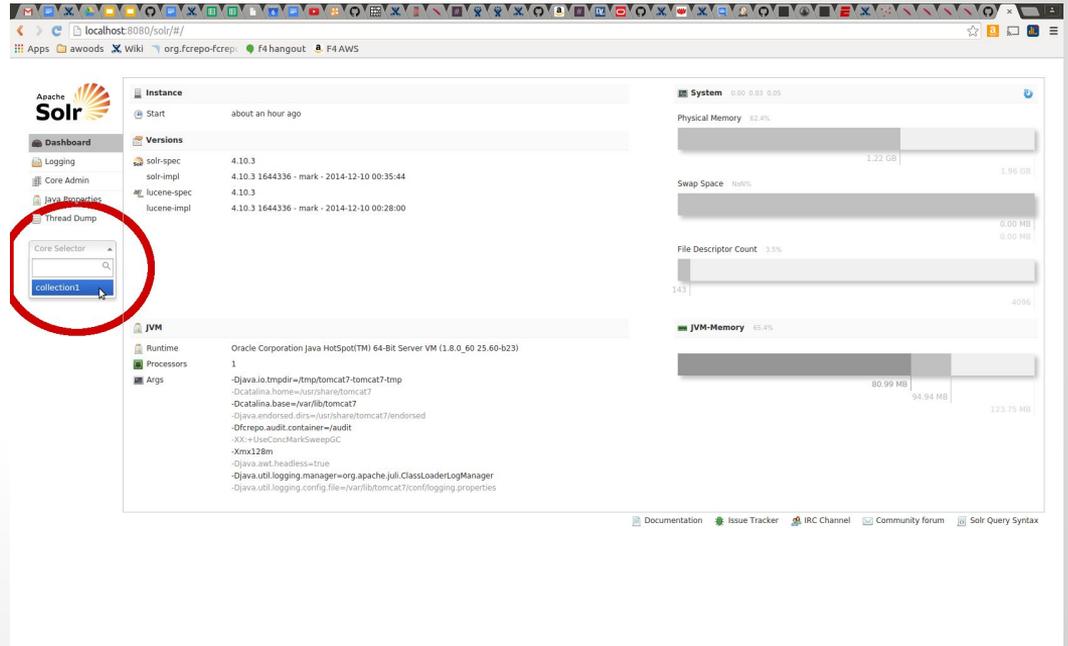
PREFIX premis: <http://www.loc.gov/premis/rdf/v1#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

```
select ?s ?d where {  
  ?s ?p <http://fedora.info/definitions/v4/audit#InternalEvent> .  
  ?s premis:hasEventRelatedObject <http://localhost:8080/fcrepo/rest/cover> .  
  ?s premis:hasEventDateTime ?d .  
  FILTER (?d > "2015-10-06T04:21:14Z"^^xsd:dateTime)  
}
```

Hands-On: Indexing in Solr

<http://localhost:8080/solr>



The screenshot displays the Apache Solr Admin UI. The left sidebar contains navigation links: Dashboard, Logging, Core Admin, Java Processes, and Thread Dump. The main content area is divided into several sections:

- Instance:** Shows the instance started 'about an hour ago'.
- Versions:** A table listing installed versions:

Component	Version
solr-spec	4.10.3
solr-impl	4.10.3 1644336 - mark - 2014-12-10 00:35:44
lucene-spec	4.10.3
lucene-impl	4.10.3 1644336 - mark - 2014-12-10 00:28:00
- System:** Displays resource usage with progress bars:
 - Physical Memory: 62.4% (1.22 GB / 1.96 GB)
 - Swap Space: 0.0%
 - File Descriptor Count: 3.3% (143 / 4096)
 - JVM-Memory: 65.4% (80.99 MB / 123.75 MB)
- JVM:** Shows runtime information for 'Oracle Corporation java HotSpot(TM) 64-Bit Server VM (1.8.0_60 25.60-b23)'. The 'Args' section lists various JVM options, including `-Djava.io.tmpdir=/tmp/tomcat7-tomcat7-tmp`, `-Dcatalina.home=/var/lib/tomcat7`, `-Dcatalina.base=/var/lib/tomcat7`, `-Djava.endorsed.dirs=/usr/share/tomcat7/endorsed`, `-Djrepro.audit.container=audit`, `-XX:+UseConcMarkSweepGC`, `-Xmx128m`, `-Djava.net.headless=true`, `-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager`, and `-Djava.util.logging.config.file=/var/lib/tomcat7/conf/logging.properties`.

A red circle highlights the 'Core Selector' dropdown menu in the left sidebar, which currently shows 'collection1' selected.

collection1

Hands-On: Indexing in Solr

The screenshot shows the Apache Solr Admin UI for a collection named 'collection1'. The left sidebar contains a navigation menu with the following items: Dashboard, Logging, Core Admin, Java Properties, Thread Dump, Overview (selected), Analysis, Dataimport, Documents, HLC, Ping, Plugins / Stats, Query (circled in red), Replication, and Schema Browser. The main content area is divided into several sections:

- Statistics:** Last Modified: about 12 hours ago, Num Docs: 7, Max Docs: 11, Heap Memory Usage: 35136, Deleted Docs: 4, Version: 21, Segment Count: 6. An 'optimize now' button is present.
- Instance:** CWD: /var/lib/tomcat7, Instance: /var/lib/tomcat7/solr/collection1, Data: /var/lib/tomcat7/solr/collection1/data, Index: /var/lib/tomcat7/solr/collection1/data/index, Impl: org.apache.solr.core.NRTCachingDirectoryFactory.
- Replication (Master):** A table showing replication status:

	Version	Gen	Size
Master (Searching)	1444106024770	7	13.38 KB
Master (Replicable)	-	-	-
- Healthcheck:** Ping request handler is not configured with a healthcheck file.
- Admin Extra:** A section for additional administrative actions.

At the bottom of the page, there are links for Documentation, Issue Tracker, IRC Channel, Community forum, and Solr Query Syntax. The browser address bar shows 'localhost:8080/solr/#/collection1' and the status bar at the bottom shows 'localhost:8080/solr/#/collection1/query'.

Hands-On: Reindexing - prep

> sudo service tomcat7 stop

> sudo rm -rf /etc/fuseki/databases/test_data/*

> sudo service tomcat7 start

Hands-On: Reindexing - check

```
select * where {  
  <http://localhost:8080/fcrepo/rest/cover> ?p ?o  
}
```

Hands-On: Reindexing

```
> curl -XPOST localhost:9080/reindexing/cover  
-H"Content-Type: application/json" -d  
'["activemq:queue:triplestore.reindex"]'
```

Hands-On: Fixity - setup

```
> sudo vi /opt/karaf/etc/org.fcrepo.camel.fixity.cfg
```

Change:

```
fixity.success=mock:fixity.success
```

To:

```
fixity.success=file:/tmp/?fileName=fixitySuccess.log&fileExist=Append
```

Hands-On: Fixity

```
> curl -XPOST localhost:9080/reindexing/cover  
-H"Content-Type: application/json" -d  
'["activemq:queue:fixity"]'
```

```
> less /tmp/fixitySuccess.log
```

```
> less /tmp/fixityErrors.log
```

Success!