APACHE SLING & FRIENDS TECH MEETUP

BERLIN, 23-25 SEPTEMBER 2013

Scaling Search in Oak with Solr

Tommaso Teofili

A look back ...

Scaling Search in Oak with Solr

- Following up last year's "Oak / Solr integration" session

Looking at last year's agenda

- What we have:
  - Search Oak content with Solr independently
  - Oak running with a Solr index
    - Embedded
    - Remote
- What we miss:
  - Solr based MicroKernel

Apache Jackrabbit Oak

- Scalable content repository
- JCR compatibility (to some degree)
- Performance especially for concurrent access
- Scalability for huge repositories (> 100M nodes)
- Support managed environments (e.g. OSGi)
- Cloud deployments

Apache Solr

- Enterprise search platform
- Based on Apache Lucene
- Full text, faceting, highlighting, etc.
- Dynamic cluster architecture
- HTTP API
- Latest release 4.4.0

Why Apache Solr

- Distributed, fault tolerant indexing / searching
- Highly configurable
  - without touching the repository
- Customizable

How it works ...

IndexEditor API

- an Editor receives info about changes to the content tree
- the Editor evaluates the status before and after a specific Oak commit
- can reject or accept the changes (by even modifying the tree itself)
- an Editor is executed right before an Oak commit is persisted
- the SolrIndexHook maps changes between statuses in the content tree to Solr documents and sends them to the Solr instance(s)

IndexEditor API – creating content

- NodeState before = builder.getNodeState();
- builder.child("newnode").setProperty("prop", "val");
- NodeState after = builder.getNodeState();
- EditorHook hook = new EditorHook(new SolrIndexEditorProvider(…));
- NodeState indexed = hook.processCommit(before, after);

QueryIndex API

- evaluate the query (Filter) cost for each available index to find the “best”
- execute the query (Filter) against a specific revision and root node state
  - internally the Filter is usually mapped to the underlying implementation counterpart
  - improved support for full text queries in Oak
- eventually view the query “plan”

QueryIndex API

- mapping Filter restrictions:
  - Property restrictions:
    - Each property is mapped as a field
      - Can use term queries for simple value matching or range queries for “first to last”
  - Path restrictions
      - indexed as strings and with special fields for parent / children / descendant matching
  - Full text expressions
      - use (E)DisMax query parser and/or fallback fields
  - NodeType restrictions TBD

Configuring the Solr index in Oak

- create an index configuration under
  - /oak:index/solrIdx
    - jcr:primaryType = oak:queryIndexDefinition
    - type = solr
      - plus some mandatory props (e.g. reindex)
    - Additional properties if want to run an embedded Solr server (more on this later)

Configuring the Solr index in Oak

- Pluggable Solr server providers and configuration
  - to allow different deployment scenarios
  - to allow custom configuration

adaptTo()

Oak Solr core bundle

- provides basic API and implementation to index and search Oak content on Solr
  - Solr implementation of IndexEditor
  - Solr implementation of QueryIndex
- allows configurable mapping between
  - property types and fields
    - e.g. all ***binaries*** should be indexed in specific field
  - filter restrictions and fields
    - e.g. path restrictions for ***children*** should hit a certain field

Oak Solr Embedded bundle

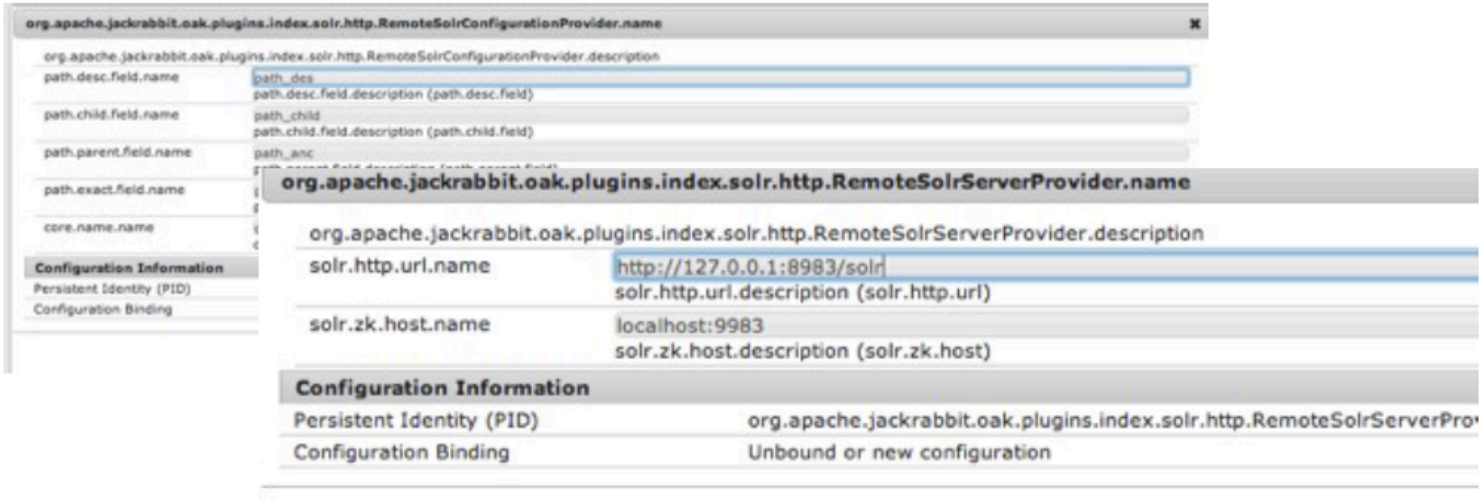
- provides support for indexing and searching on an embedded Solr instance
  - running inside the Oak repository
  - configuration can be done
    - via the repository
      - stored in the index definition node
    - via OSGi

Oak Solr Remote bundle

- provides support for indexing and searching on remote Solr instances
  - single Solr instance
  - distributed / replicated Solr cluster
  - SolrCloud deployments
  - configuration is done via OSGi

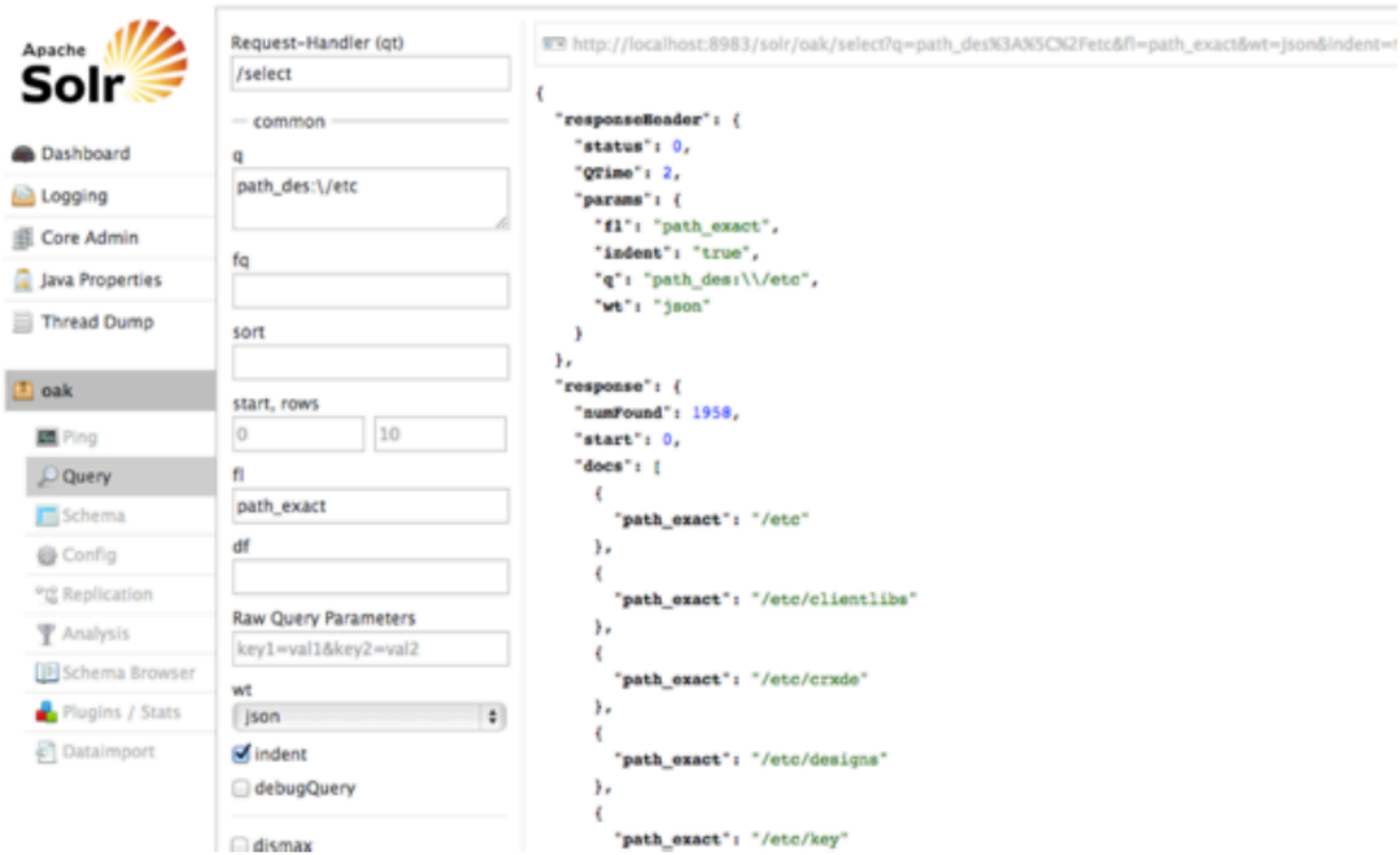
OSGi platform running on Oak with Solr

- Star instance with Oak repository
- Add a bunch of bundles for Solr
  - oak-solr-core, oak-solr-remote, zookeeper, servicemix.bundles.solr-solrj, etc.
- Configure the Solr instance
- Configure oak-solr-remote providers via OSGi



See it in action ...

Solr index populated with Oak content



What needs to be done

- Easy OSGi deployment
- Move common configuration stuff in oak-solr-core
- Leverage new full text expression API
- Solr MK?