JCR Deep Dive

Tyler Maynard
AEM DEVELOPER

@TylersDesk www.tylermaynard.com

Overview

JCR model

JCR event modelling

Querying index

Indexing tools

Query syntax

JCR Model

How JCR differs from RDB

Hierarchical

Flexible

Java API

Queries and full text search

JCR Features

Query via SQL, JQOM, and Xpath

Export/Import (XML)

Referential Integrity

Authentication, Access Control & Versioning

Observation

Locking and transactions (JTA)

Understanding David’s Model

Data first,
structure later - maybe

Drive the content hierarchy

Clone(), merge(), and
update()

Beware of same name siblings

Understanding David’s Model Continued

References considered harmful

Files are files are files

IDs are evil

Content Services of JCR

Author based versioning

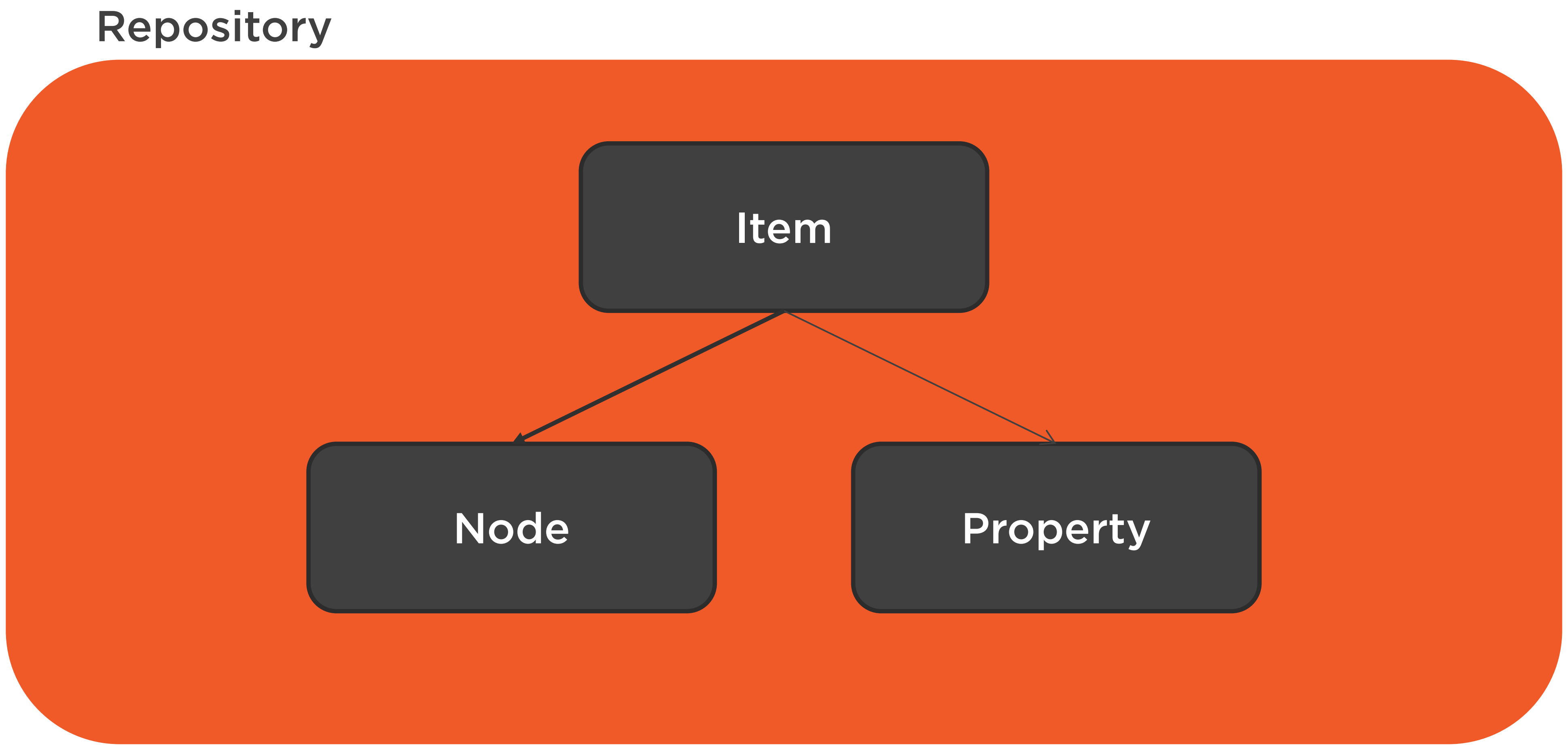
Full-text searching

Fine-grained access control

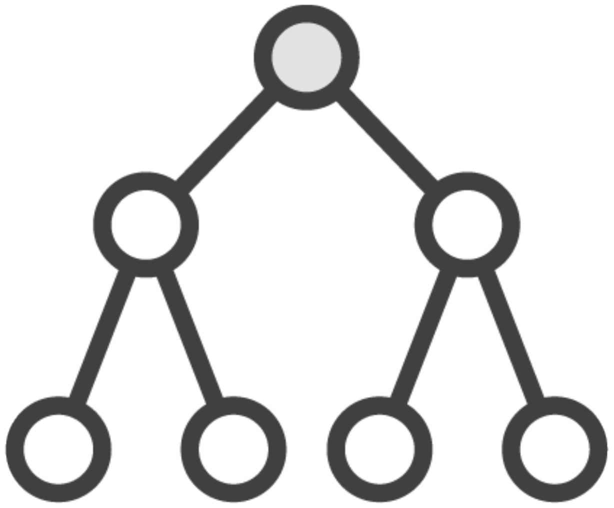
Content categorization

Content event monitoring

Structure of the JCR



Nodes



One or more types associated with them

Point to other nodes

Version-able

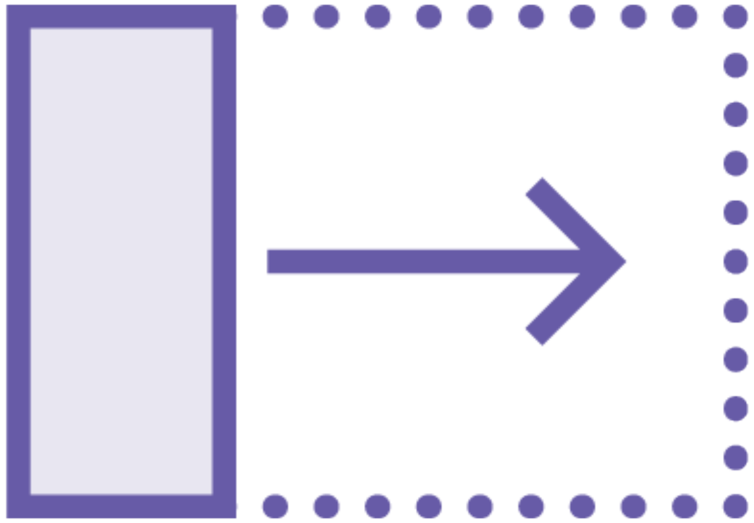
Properties

Single or multi-valued

12 possible types

Node Types

jcr:primaryType



Mixin node types

Node Type Definitions

Stored in the form of node type definitions

Javax.jcr.nodetype.NodeType

Javax.jcr.Node.getPrimaryNodeType()

Set mandatory attributes

Node Type Inheritance

Property definitions

Child node definitions

Other attributes, such as ‘isMixin’

Event Modelling

Event Model Overview

Item added

Item changed

Item moved

Item removed

The Event Object & Event Types

Event object

Event.getType()

Types of Events

Node added

Node moved

Node removed

Property added

Property removed

Property changed

Persist

Event Information

Information	Accessed By
Event path	`String Event.getPath()`
Identifier	`String Event.getIdentifier()`
Information map	`java.util.Map Event.getInfo()`

Node Added or Node Removed Events

Event Type

`NODE_ADDED or NODE_REMOVED`

Method	Returns
`Event.getPath()`	Absolute path of the node that was added or removed
`Event.getIdentifier()`	Identifier of the node that was added or removed
`Event.getInfo()`	Empty Map object

Node Moved Events

Event Type
`NODE_MOVED`

Method	Returns
`Event.getPath()`	**Absolute path of the destination of the move**
`Event.getIdentifier()`	**Identifier of the node that was moved**
`Event.getInfo()`	**Map containing parameters information from the method that caused the event**

Property Added, Changed or Removed Events

Event Type
`PROPERTY_ADDED, PROPERTY_CHANGED, or PROPERTY_REMOVED`

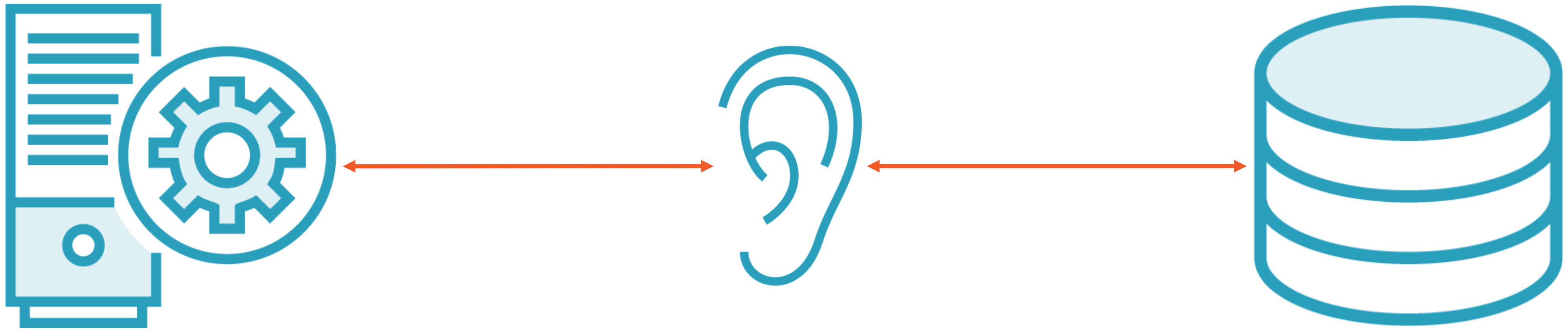
Method	Returns
`Event.getPath()`	Absolute path of the property that was added, changed, or removed
`Event.getIdentifier()`	Identifier of the parent node of the property that was added, changed, or removed
`Event.getInfo()`	Empty Map object

Persist Events

Event Type
`PERSIST`

Method	Returns
`Event.getPath()`	**null**
`Event.getIdentifier()`	**null**
`Event.getInfo()`	**empty Map**

Asynchronous Observation



Application
(AEM/Servlet/Logic)

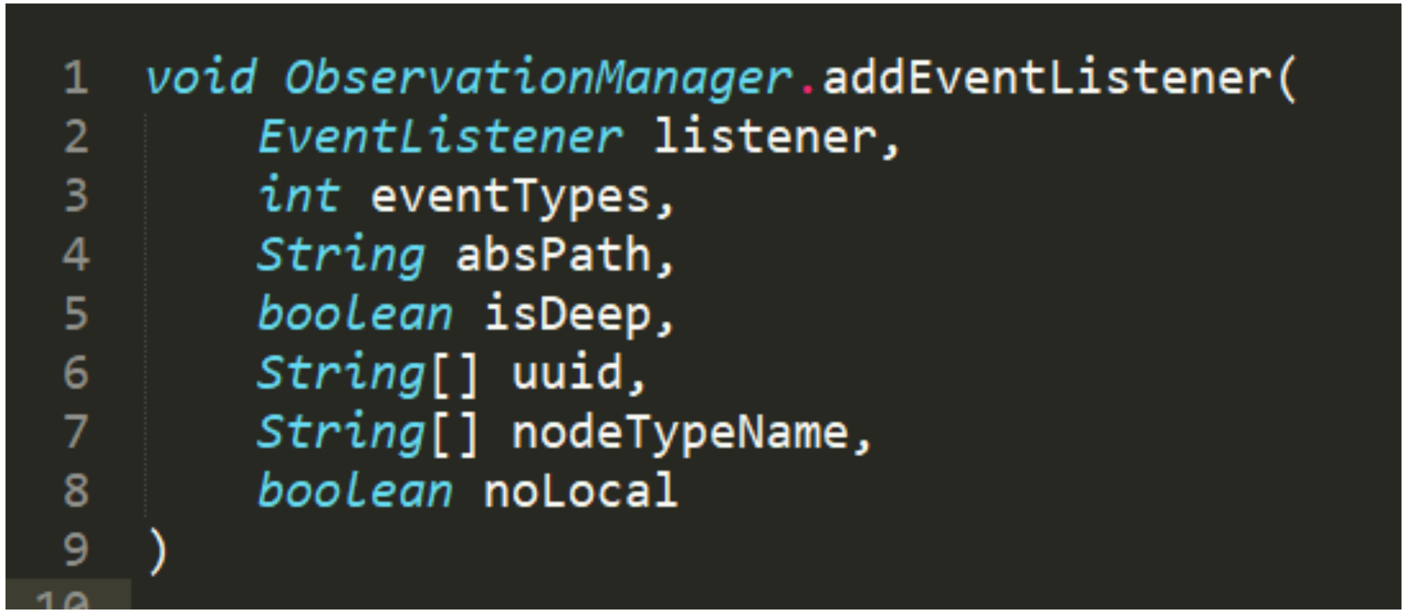
eventListener

Stream of events

Understanding Asynchronous Observation

```
ObservationManager Workspace.getObservationManager();
```

Adding an Event Listener



```
void ObservationManager.addEventListener(
    EventListener listener,
    int eventTypes,
    String absPath,
    boolean isDeep,
    String[] uuid,
    String[] nodeTypeName,
    boolean noLocal
)
```

```
void EventListener.onEvent(EventIterator events)
```

Re-registration of Listeners & Event Iterator

```
Event EventIteratornextEvent()
```

Listing Event Listeners

Method	Purpose
`EventListenerIterator.nextEventListener()`	**EventListener-specific next method**
`ObservationManager.removeEventListener(EventListener listener)`	**Removing Event Listeners**
`ObservationManager.setUserData(String userData)`	**Set the user data**

Create an Observation Listener

Demo

Create TitlePropertyListener class

Change jcr:title property