Hydra Curriculum (2010)

About this Document

This was the working document for a Hydra Curriculum circa 2010. Most of this content became part of the HydraCamp curriculum.

Why a curriculum?

In the effort to create digital repositories, libraries rarely have the luxury of hiring new staff who are both fluent in the necessary technology and cognizant of the particular concerns and responsibilities of libraries. The technology skills required for the implementation of a [digital library | institutional repository | electronic thesis and dissertation solution] are so new that few training resources are available. This document attempts to provide a resource for training staff in the technology skills necessary for implementing a digital repository. We list the technologies and the problem solving skills required to work effectively with each technology, along with a practical example of how each can be used in the creation and running of a repository.

Who is this intended for?

Hydra Training Curriculum (2010)

Goal: Build a curriculum similar to the WaSP InterACT Curriculum.

Guidelines

Focus on pragmatic info, ie. Don't spend too much time on history or alternative technologies.
People are here to learn what they need to know in order to get the job done effectively.
Have examples beforehand for group excersizes/challenges
Ask lots of questions (to retain engagement)
Have sample content

Framing the Issue

Data Deluge - ie. why are we all suddenly dealing with datasets
Responsibilities of Curators

- allow creators & consumers to define the terminologies, schems, etc. Capture their info as they express it rather than attempting to force your concepts on them.

Data Integrity & Trust-Enabled Repositories
Browse vs. Search vs. Facets
User-Centric Development & Design

Collaborative Development

We're in this Together
Project Mission Statements
Distributed Version Control -- Git

- forking, cloning, committing, pushing
- submitting pull requests
- pulling upstream changes
- creating good commits (granular commits with descriptive log messages)

Project Trackers & Writing a good Ticket

- stories
- bugs
- documentation requests

Documentation

- wikis
- code documentation
- installation docs
- tutorials

Stories vs Technical Features
irc & mailing lists
Consistent Development Environments -- RVM, Dependency management, etc.

Recommended Reading:

- Creating Open Source Software

Solr

1. Figure out what your discoverability goals are
2. Meeting your discoverability goals
3. Evaluating whether you've met your discoverability goals

Difference between a Relational Database and a Search Index
Understanding why you index the same thing multiple times
What is a facet? Facet count?
Reading a solr document in xml, json, etc
Tools & Resources (ie. index inspection tools)

Ruby on Rails / Blacklight

Installing Blacklight
MVC & Why it's useful

- controllers setting variables for views
- naming conventions (ie. post.rb, posts_controller.rb, views/posts/[index, show, view, edit, new]

Rails app directory structure

- /app
- /config
- /vendor/plugins

Engines & local overrides
RESTful design & routing & resource-oriented design
Testing -- Eat your vegetables
Rails environments, yml files, jetty instances, production servers and staging servers

Introduction to Blacklight:

.rvmrc

Solr Documents
solr schema
* stored vs unstored
* copyfields

Tokenizers
solr config
Solr search handlers

blacklight_config
CatalogController
Blacklight views
render_document_partial

Fedora

What Fedora is
Fedora Admin Client
Fedora Data Model

- What a Fedora Object is
- What a datastream is
- RELS-EXT

Storing files in Fedora
Adding metadata to Fedora Objects
Content Models ... compound vs atomistic ... mixins
Fedora CMA & Fedora ECM

Fedora storage model
Contrast with File System
Contrast with RDBMS

Triplestore & Querying it ... subject-predicate-object

ActiveFedora

Defining a Model
Manipulating Datastreams
Manipulating Relationships
Command line tour
Batch Operations ... writing a batch script

Learning Outcome:
You will be able to:

- write a script that will iterate through a directory of files & create fedora objects for each of them
- write a script that runs a search in Fedora and iterates through the results, updating the metadata on each of them

Hydra Rails Integration

Learning Outcome:
Make a web application that creates, displays, edits & saves Fedora objects and displays operations on DC metadata. This application will lend itself to being reformatted to use additional metadata schemas.

XML Metadata

Essential (Bibliographic) Descriptive Metadata
- Dublin Core
- MODS
- MARC XML

Specialized Descriptive Metadata
- TEI (Marking up Text Documents)
- EAD (Archival Description)
- VRE (Describing Visual Resources)
- PBCore (Describing Recorded/Broadcast Materials)

Technical Metadata
- Hydra rightsMetadata
- XACML
- Fedora AUDIT datastream

Structural XML Metadata
- METS
- FOXML

Vocabularies
- LoC Subject Headings
- Getty Vocabularies (TGN, AAT, ULAN, CONA)
- Pubmed MESH Headings

OM

Learning Outcome:
Write an OM Terminology that allows you to use a custom vocabulary to work with an arbitrary xml schema and use that terminology to parse xml, look up values and update the xml from the command line & index that xml into solr.

Solrizer

Refine & customize the way an OM Document is indexed into solr
Index a Fedora object into solr according to multiple applicable AF models
Index all of the objects from a Fedora repository into Solr
Trigger Solrizer from the command line, from coe, or from a JMS listener

Assessment & Metrics for Success

End Users, External Audits, Submission Statistics

Learning Outcome:
Plan and use metrics for assessing whether your repository project is successful

Examples:

- # of objects in repository
- # of submissions in 3 months
- # of submitters in 3 months
- consumption statistics
- user feedback