



Samvera Connect 2018

Introduction to Valkyrie

10/09/2018 • 9am-noon • Esmé Cowles & Adam Wead

<http://bit.ly/valkyrie-intro>

Samvera fosters Open Culture and Open Knowledge
through the work we do and
by the way we work together

samvera-wiki/code+of+conduct



Anti-Harassment Policy

- See [samvera-wiki/Anti-Harassment+Policy](https://samvera-wiki/anti-harassment+policy)
- Event organizers, Steering Committee Members, and [Helpers](#) are available if you feel threatened or unsafe in any way

Contact us if you have any concerns or questions.



Agenda



9:00 - 9:30	Introductions and Overview
9:30 - 10:00	Hands-On: Command-Line Demo
10:00 - 10:30	Valkyrie App Demos: Cho & Figgy
10:30 - 10:40	Break
10:40 - 11:10	Hands-On: Rails App Demo
11:10 - 11:30	Backends, ChangeSets, New Features, Hyrax, etc.
11:30 - 12:00	Questions and Comments

Introductions



- Esmé Cowles, Princeton
- Adam Wead, Penn State
- Attendees
 - Name and Institution
 - Why are you interested in Valkyrie?
 - What backend, use case, or other topic do you want to hear about?

What Is Valkyrie?



- Gem: <https://github.com/samvera-labs/valkyrie>
- Written by: Data Mapper Working Group and other collaborators
- APIs for working with metadata and files
 - replacement for ActiveFedora
- Goal: Allow Samvera applications to use different backends to store their metadata and files, but still share application code.

Workshop Setup

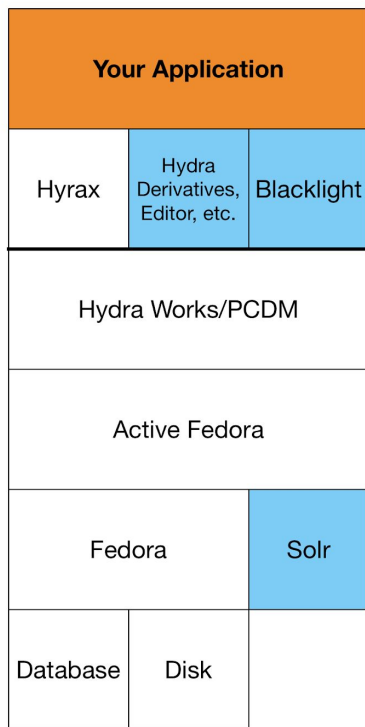


- `git clone https://github.com/escowles/vdemo`
- `cd vdemo`
- `bundle install`
- Edit `config/database.yml` to match local database settings, if needed
- `bundle exec rake db:create:all`
- `bundle exec rake db:migrate`
- `bundle exec rake db:migrate RAILS_ENV=test`
- `bundle exec rails c`

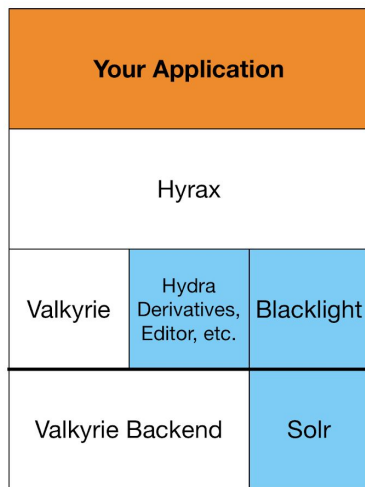
Overview

Where does Valkyrie fit?

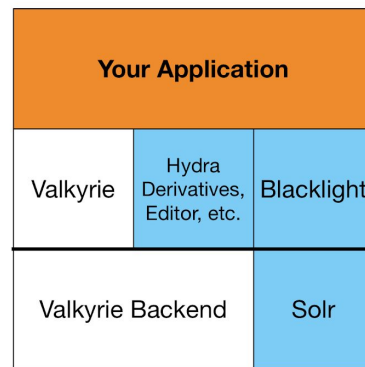
Hyrax 1 & 2 Architecture



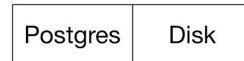
Hyrax 3 Architecture



Valkyrie Architecture



Production Valkyrie Backend Options



Why Was Valkyrie Created?



<https://github.com/samvera-labs/valkyrie/wiki/Frequently-Asked-Questions>

- Allow using technologies which fit your use cases, timelines, and opinions.
- Provide a common interface to multiple databases to continue working together and sharing code.

Active Record/Active Fedora Pattern



- "Thick" models, that include validation and other logic
- Models are the primary API
 - *Model.find, object.save, object.delete, etc.*
- Persistence choices are hidden by the models
 - ActiveFedora: not always clear what uses Fedora, Solr, or both


Data Mapper Pattern



- "Thin" models, validation and other logic are handled by other classes
- Mappers are the primary API
 - Models are arguments to methods that query, save, delete, etc.
- Persistence choices are modeled explicitly

Comparison to ActiveFedora



	ActiveFedora	Valkyrie
Find a Record	<code>Book.find(id)</code>	<code>adapter.query_service.find_by(id: id)</code>
Save	<code>book.save</code>	<code>adapter.persister.save(resource: book)</code>
Delete	<code>book.destroy</code>	<code>adapter.persister.delete(resource: book)</code>
Save to Fedora/Solr	<code>book.save</code>	<code>combined_adapter.persister.save(resource: book)</code>
Save only to Fedora	?	<code>fedora_adapter.persister.save(resource: book)</code>
Save only to Postgres		<code>postgres_adapter.persister.save(resource: book)</code>
Migrate Schema	"I guess I'll write a script?"	<code>book = old_adapter.query_service.find_by(id: id)</code> <code>new_adapter.persister.save(resource: book)</code>

Valkyrie Core Concepts



- Resource: "thin" model
- Metadata Adapter: working with metadata
 - Query Service: read
 - Persister: create, update, and delete
- Storage Adapter: working with files
- Change Set: update and validation logic
- Decorator: display logic

External Libraries Used By Valkyrie



- Resource:
 - Dry::Struct: <https://dry-rb.org/gems/dry-struct/>
 - Dry::Types: <https://dry-rb.org/gems/dry-types/>
- Change Set:
 - Reform::Form: <https://github.com/trailblazer/reform>
- Decorator:
 - Draper::Decorator: <https://github.com/drapergem/drapeer>

Resources



vdemo/app/models/armor.rb

```
class Armor < Valkyrie::Resource
  attribute :title, Valkyrie::Types::String
  attribute :member_ids, Valkyrie::Types::Array
end
```

Resources: Data Types



Valkyrie supports the following data types:

- String
- Integer
- Float
- DateTime
- RDF::Literal
- RDF::URI
- Valkyrie::ID (internal relationships)
- Nested Objects

Registering Metadata and Storage Adapters



vdemo/config/initializers/valkyrie.rb

```
Rails.application.config.to_prepare do
  Valkyrie::MetadataAdapter.register(
    Valkyrie::Persistence::Memory::MetadataAdapter.new, :memory)

  Valkyrie::MetadataAdapter.register(
    Valkyrie::Persistence::Postgres::MetadataAdapter.new, :postgres)

  Valkyrie::StorageAdapter.register(
    Valkyrie::Storage::Memory.new, :memory)

  Valkyrie::StorageAdapter.register(
    Valkyrie::Storage::Disk.new(base_path: 'tmp/files'), :disk)
end
```

Configuring Metadata and Storage Adapters



vdemo/config/valkyrie.yml

development:

metadata_adapter: postgres

storage_adapter: disk

test:

metadata_adapter: memory

storage_adapter: memory

Working With Metadata and Storage Adapters



- Default values
 - `Valkyrie.config.metadata_adapter`
 - `Valkyrie.config.storage_adapter`
- Loading by label
 - `Valkyrie::MetadataAdapter.find(:postgres)`
 - `Valkyrie::StorageAdapter.find(:disk)`

Persister: Saving and Deleting Resources



- `save(resource: r)` — creates or updates a single resource
- `save_all(resources: [r1, r2])` — creates or updates multiple resources
- `delete(resource: r)` — deletes a single resource

Queries: Identifiers and Types



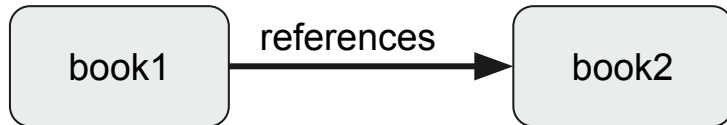
<https://github.com/samvera-labs/valkyrie/wiki/Queries>

- `find_by(id: '123')` — loads a single resource by id
- `find_by_many_ids(ids: ['123', '456'])` — loads multiple resources by id
- `find_all` — loads all resources
- `find_all_of_model(model: Armor)` — loads all resources of a given class
- `find_by_alternate_identifier(alternate_identifier: 'asdf')`
 - loads a single resource by an "alternate" id (old system id, ARK/DOI/etc.)

Queries: References

<https://github.com/samvera-labs/valkyrie/wiki/Queries>

- `find_references_by(resource: res, property: prop)`
 - load resources that link from *res* using property *prop*
- `find_inverse_references_by(resource: res, property prop)`
 - load resources that links to *res* using property *prop*



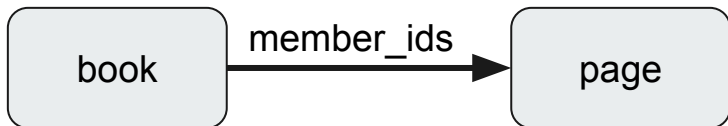
```
find_references_by(resource: book1, property: 'references') => book2
```

```
find_inverse_references_by(resource: book2, property: 'references') => book1
```

Queries: Membership

<https://github.com/samvera-labs/valkyrie/wiki/Queries>

- `find_members(resource: r, model: Helmet)`
 - loads resources linked from `member_ids`, optionally limited by `model`
- `find_parents(resource: r)`
 - load resources that link to this resource from `member_ids`



```
find_members(resource: book) => page  
find_parents(resource: page) => book
```

Queries: Custom



<https://github.com/samvera-labs/valkyrie/wiki/Queries>

- Literally anything you want
 - Try out new features
 - Customize to your local needs
 - Do things more efficiently for the backends you use
- But: can make your application work only with the backends you use

Storage Adapter: Saving and Loading Files



- `upload(file: f, resource: res, original_filename: 'foo.txt')`
— store a file *f*
- `find_by(id: '123')` — load a file by id
- `delete(id: '123')` — delete a file by id

Working With Files



- `read` — read the contents of a file into memory
- `stream` — read the contents of a file as a stream
- `rewind` — reset the stream
- `size` — get the file's size in bytes
- `disk_path` — access as a file on disk (streamed files will be cached locally)
- `checksum(digests: [Digest::MD5.new])`
 - calculate checksums of a file using the given algorithm
- `valid?(size: 1722, digests:{md5: "4ead20c186eaf2f7c09d6627ab7c0102"})`
 - validate that size and checksums match provided values

Hands-On: Command-Line Demo

Command-Line Demo



<https://github.com/escowles/vdemo/wiki/Demo>

Valkyrie App Demos

Valkyrie App Demos



- Cho: <https://github.com/psu-libraries/cho>
- Figgy: <https://github.com/pulibrary/figgy>
 - <https://figgy.princeton.edu/>

Break

Please return by 10:40!



Hands-On: Rails App Demo

Hands-On: Rails App Demo



vdemo: <http://localhost:3000/>

Ideas



- Add a field
- Add a new model
- More validation in ChangeSets
- Create a FileMetadata nested class to hold file timestamp, filename, etc.
- Use a different backend
- Add styling or Javascript
 - e.g., Helmets could have multiple creators...

**Backends, ChageSets, New
Features, Hyrax, etc.**

Backends



<https://github.com/samvera-labs/valkyrie/wiki/Supported-Backends>

- Core Metadata Adapters
 - Memory
 - Fedora
 - PostgreSQL
 - Solr
- External
 - ActiveRecord
 - AWS CloudSearch
 - AWS DynamoDB
 - Redis
 - Sequel/PostgreSQL
- Core Storage Adapters
 - Memory
 - Fedora 4
 - Disk
- External
 - None yet

Change Sets



1. Like Form Objects
2. Expected to use them to persist objects.
3. Can hold attributes which aren't to be persisted, but you can trigger logic off of.
4. Can handle converting values to/from an array.
5. Manage validations
6. Change sets in the wild:
 - a. https://github.com/psu-libraries/cho/blob/master/app/cho/work/submission_change_set.rb

Change Set Persisters



1. Can combine multiple persisters, ex. Postgres and Solr
2. Wrap additional logic to updates, creates, deletes, etc.
 - a. Minting ARKs
 - b. Using transactions
 - c. Deleting child objects
3. Change set persister in the wild:
 - a. https://github.com/psu-libraries/cho/blob/master/app/valkyrie/change_set_persister.rb

Deployment



1. Standard Rails practices apply
 - a. Capistrano
 - b. Chef
 - c. Others
2. Persister support, ex. Postgres and Solr, or others

New Features



- Optimistic Locking: prevent multiple updates from overwriting each other
- Ordered Properties: order titles, creators, or any property
- Singular Values: properties can be singular, Sets or Arrays

Hyrax



- Much work went into the valkyrie branch of Hyrax
 - But not backwards-compatible with Hyrax 1 / 2 (data migration required)
 - Conflicts after Collection Extension merged
- New work on a backwards-compatible approach
 - Breakout session on Wednesday to review past work and plan how to move this work forward

Questions & Comments

Thank You



- Esmé Cowles, @escowles
- Adam Wead, @awead