



Intro to Valkyrie

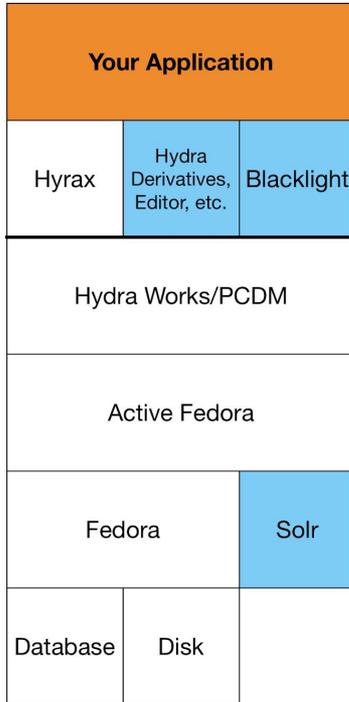


What's Valkyrie?

1. Gem written by the Data Mapper Working Group and other collaborators to allow Samvera applications to use different backends to store their metadata and files, but still share application code.
2. <https://github.com/samvera-labs/valkyrie>

Where does Valkyrie fit?

Hyrax 1 & 2 Architecture



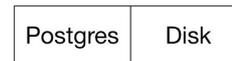
Hyrax 3 Architecture



Valkyrie Architecture



Production Valkyrie Backend Options





Why?

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

1. Allow partners to choose technologies which fit their use cases, timelines, and opinions.
2. Provide a common interface to multiple databases to ease processes like reindexing and data migration.



Active Record Pattern

“An object that wraps a row in a database table or view, encapsulates the database access, and adds domain logic on that data.” -- Martin Fowler

The object you have looks the same as what's in the database, and you can interact with the database through that object.

`ActiveRecord::Base.find`, `book.save`, `book.delete`, etc.



Data Mapper Pattern

“A layer of Mappers that moves data between objects and a database while keeping them independent of each other and the mapper itself.” -- Martin Fowler

The object is decoupled from persistence. Instead, you pass objects to Mappers in order to coordinate with the database.

Valkyrie::MetadataAdapters are “Mappers.”

Comparison



	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 to Postgres		<code>postgres_adapter.persister.save(resource: book)</code>
Migrate Schema	“How do I write this script?”	<code>book = old_adapter.query_service.find_by(id: id)</code> <code>new_adapter.persister.save(resource: book)</code>

Valkyrie::Resource

```
class Postcard < Valkyrie::Resource
  attribute :id, Valkyrie::Types::ID.optional
  attribute :title, Valkyrie::Types::Set
  attribute :author, Valkyrie::Types::Set
end
```

Metadata Adapters - Registration



Metadata Adapters are registered using a short name

```
Valkyrie::MetadataAdapter.register(  
  Valkyrie::Persistence::Memory::MetadataAdapter.new,  
  :memory  
)
```

Metadata Adapters - Usage



Metadata Adapters - Data Types



Valkyrie supports the following data types:

1. Integers
2. Strings
3. RDF::URI
4. RDF::Literal
5. Valkyrie::ID (Internal relationships)
6. DateTime

Metadata Adapters - Persister

```
valkyrie_workshop master % rails c
Running via Spring preloader in process 29034
Loading development environment (Rails 5.1.4)
[1] pry(main)> adapter = Valkyrie::MetadataAdapter.find(:memory)
=> #<Valkyrie::Persistence::Memory::MetadataAdapter:0x007facdfce9508>
[2] pry(main)> postcard = Postcard.new
=> #<Postcard internal_resource="Postcard" created_at=nil updated_at=$
nil id=nil title=[] author=[]>
[3] pry(main)> created_postcard = adapter.persister.save(resource: p$
stcard)
=> #<Postcard internal_resource="Postcard" created_at=2017-11-04 15:$
0:48 UTC updated_at=2017-11-04 15:10:48 UTC id=#<Valkyrie::ID:0x007f$
cdfa4e208 @id="582c7021-9eed-48df-9f01-e119e7c36e9e"> title=[] autho$
=[]>
[4] pry(main)>
```

Metadata Adapters - Find By ID



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

```
[4] pry(main)> adapter.query_service.find_by(id: created_postcard.id)
=> #<Postcard internal_resource="Postcard" created_at=2017-11-04 15:12:44 UTC updated_at=2017-11-04 15:12:44 UTC id=#<Valkyrie::ID:0x007facdcdf39b0 @id="620063dc-c196-4889-998f-70f44c92d0aa"> title=[] author=[]>
[5] pry(main)> █
```

Metadata Adapters - Find All

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

Metadata Adapters - Find All of Model



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

Metadata Adapters - Find Members

```
[1] pry(main)> adapter = Valkyrie::MetadataAdapter.find(:memory)
=> #<Valkyrie::Persistence::Memory::MetadataAdapter:0x007fe2ece226c0>
[2] pry(main)> child = adapter.persister.save(resource: Postcard.new)
=> #<Postcard internal_resource="Postcard" created_at=2017-11-04 15:19:03 UTC updated_at=2017-
d801ab29947"> title=[] author=[] member_ids=[]>
[3] pry(main)> parent = adapter.persister.save(resource: Postcard.new(member_ids: child.id))
=> #<Postcard internal_resource="Postcard" created_at=2017-11-04 15:19:09 UTC updated_at=2017-
a638745a3bd"> title=[] author=[] member_ids=[#<Valkyrie::ID:0x007fe2ecff0948 @id="20c96086-13f
[4] pry(main)> children = adapter.query_service.find_members(resource: parent)
=> [#<Postcard internal_resource="Postcard" created_at=2017-11-04 15:19:03 UTC updated_at=2017-
3d801ab29947"> title=[] author=[] member_ids=[]>]
[5] pry(main)> █
```

Metadata Adapters - Find Parents



Metadata Adapters - Find References



Metadata Adapters - Find Inverse References



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. Manages validations

Storage Adapters - Registration

```
Valkyrie::StorageAdapter.register(  
  Valkyrie::Storage::Disk.new(  
    base_path: Rails.root.join("tmp", "files"),  
    file_mover: FileUtils.method(:cp)  
  ),  
  :disk  
)
```

Storage Adapters - Upload

```
[6] pry(main)> file = File.open(Rails.root.join("public", "404.html"))
=> #<File:/vagrant/public/404.html>
[7] pry(main)> resource = metadata_adapter.persister.save(resource: Postcard.new)
W, [2017-11-06T15:46:50.051371 #2963] WARN -- : The Solr adapter is not meant to persist new resources, but is now generating an ID.
=> #<Postcard internal_resource="Postcard" created_at=2017-11-06 15:46:50 UTC updated_at=2017-11-06 15:46:50 UTC id=#<Valkyrie::ID:0xba9b1bc @id="disk:///vagrant/tmp/files/9a/33/71/9a3371dd841d4e8c8d077763a1f6a3077763a1f6a337/404.html">
member_ids=[] related_objects=[]>
[8] pry(main)> uploaded_file = storage_adapter.upload(file: file, resource: resource, original_filename: "404.html")
=> #<Valkyrie::StorageAdapter::File id=#<Valkyrie::ID:0xba9b1bc @id="disk:///vagrant/tmp/files/9a/33/71/9a3371dd841d4e8c8d077763a1f6a3077763a1f6a337/404.html">>
```

Storage Adapters - Pull with ID



```
[10] pry(main)> reloaded_file = Valkyrie::StorageAdapter.find_by(id: uploaded_file.id)

=> #<Valkyrie::StorageAdapter::File id=#<Valkyrie::ID:0xba2d25c @id="disk:///vagrant/tmp/files/9a/33/71/9a3371dd841d4e8c8d077763a1f6a337/404.html"> io=#<File:/vagrant/tmp/files/93a/33/71/9a3371dd841d4e8c8d077763a1f6a337/404.html>>
```

Storage Adapters - Checksums

```
[11] pry(main)> reloaded_file.checksum(digests: [Digest::MD5.new])
=> ["4ead20c186eaf2f7c09d6627ab7c0102"]
[12] pry(main)> reloaded_file.valid?(digests: { md5: "4ead20c186eaf2f7c09d6627ab7c0102" })
=> true
```

Storage Adapters - Delete

```
[13] pry(main)> storage_adapter.delete(id: reloaded_file.id)
=> ["/vagrant/tmp/files/9a/33/71/9a3371dd841d4e8c8d077763a1f6a337/404.html"]
[14] pry(main)> storage_adapter.find_by(id: reloaded_file.id)
Valkyrie::StorageAdapter::FileNotFound: Valkyrie::StorageAdapter::FileNotFound
from /home/vagrant/.rvm/gems/ruby-2.3.3/bundler/gems/valkyrie-894e655c9c8e/valkyrie/lib/v
alkyrie/storage/disk.rb:39:in `rescue in find_by'
```

Questions?

