Digital Commonwealth: A Repository Serving the Libraries of Massachusetts

digitalcommonwealth.org

Digital Commonwealth is a consortium of over 140 institutions that provides resources and services to support the creation, management, and dissemination of cultural heritage materials held by libraries, museums, historical societies, and archives across the state of Massachusetts.

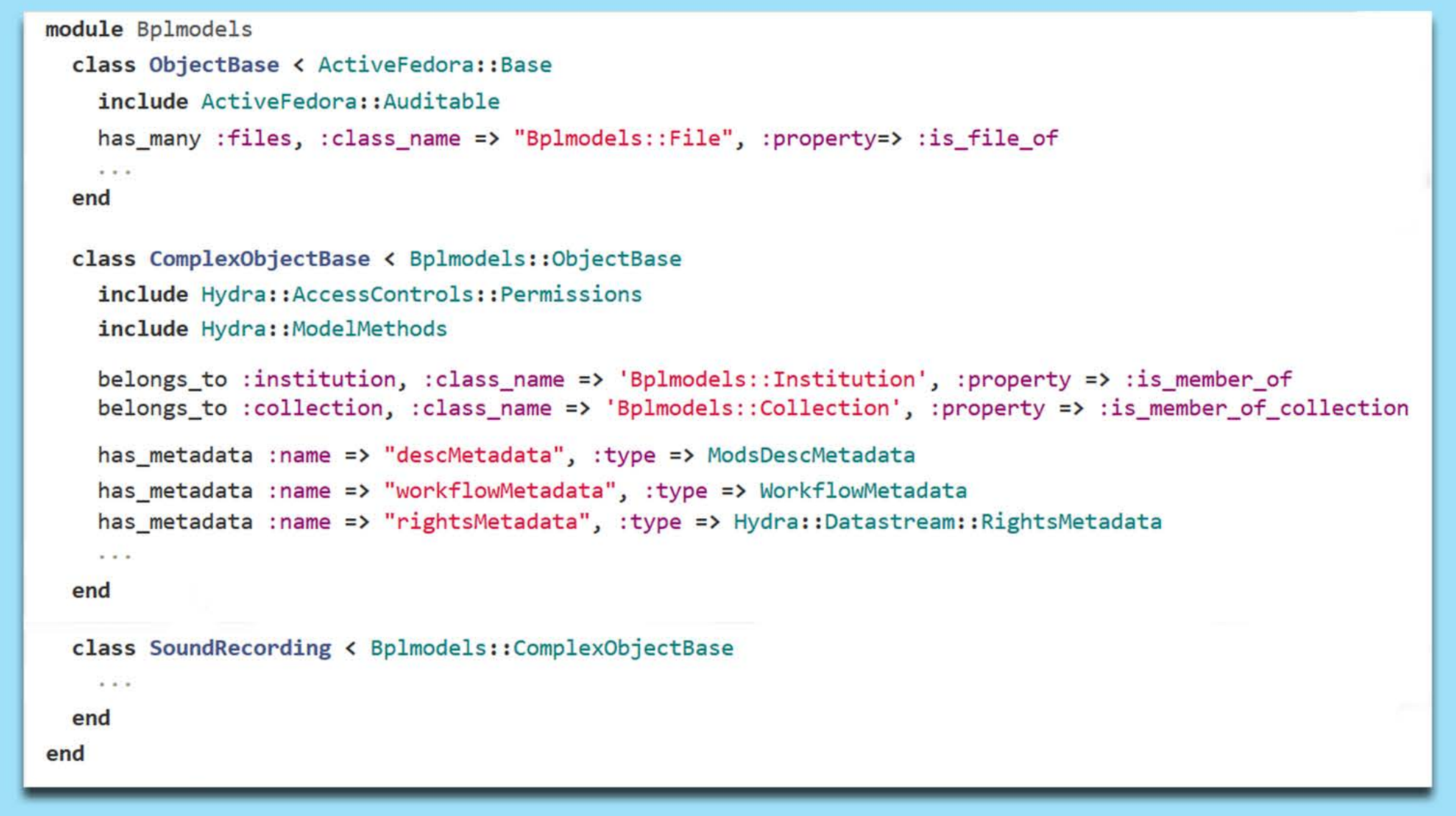
Digital Commonwealth is a repository that provides administrative control over digital objects and metadata for items from member institutions. The majority of these libraries don't have the resources to maintain their own repository, so this system provides a vital service they can depend on to manage and preserve their digital assets.

Digital Commonwealth is a discovery tool that offers access to thousands of images, documents, and sound recordings from Massachusetts libraries for researchers, students, and the general public. The application features a number of noteworthy customizations and additions o Blacklight.

Digital Commonwealth is a DPLA Service Hub that aggregates digital objects and metadata records from a diverse array of Massachusetts cultural heritage institutions in order to provide the Digital Public Library of America with unique metadata records through a single ata feed.

Content Modeling

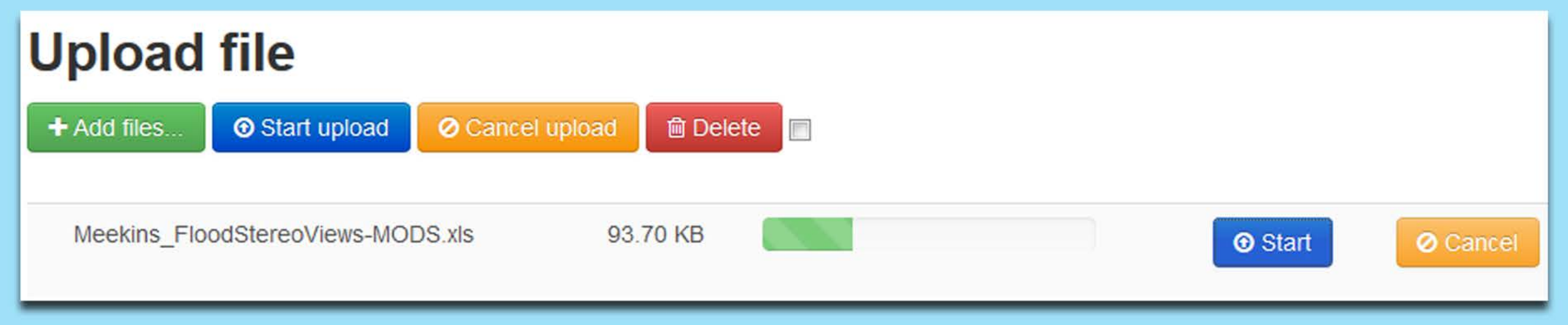
In order to facilitate sharing content models between Hydra codebases for multiple applications, a completely separate and independent Rails engine to express content models has been developed. This allows pre-configured content models to easily be re-used simply by installing a gem. We've also taken an OOP-inspired approach to modeling content, using specific content type classes that inherit from a set of more generic content models.



```ruby
module Bplmodels
  class ObjectBase < ActiveFedora::Base
    include ActiveFedora::Auditable
    has_many :files, :class_name => "Bplmodels::File", :property=> :is_file_of
    ...
  end

  class ComplexObjectBase < Bplmodels::ObjectBase
    include Hydra::AccessControls::Permissions
    include Hydra::ModelMethods

    belongs_to :institution, :class_name => 'Bplmodels::Institution', :property => :is_member_of
    belongs_to :collection, :class_name => 'Bplmodels::Collection', :property => :is_member_of_collection

    has_metadata :name => "descMetadata", :type => ModsDescMetadata
    has_metadata :name => "workflowMetadata", :type => WorkflowMetadata
    has_metadata :name => "rightsMetadata", :type => Hydra::Datastream::RightsMetadata
    ...
  end

  class SoundRecording < Bplmodels::ComplexObjectBase
    ...
  end
end
```

Batch Ingest

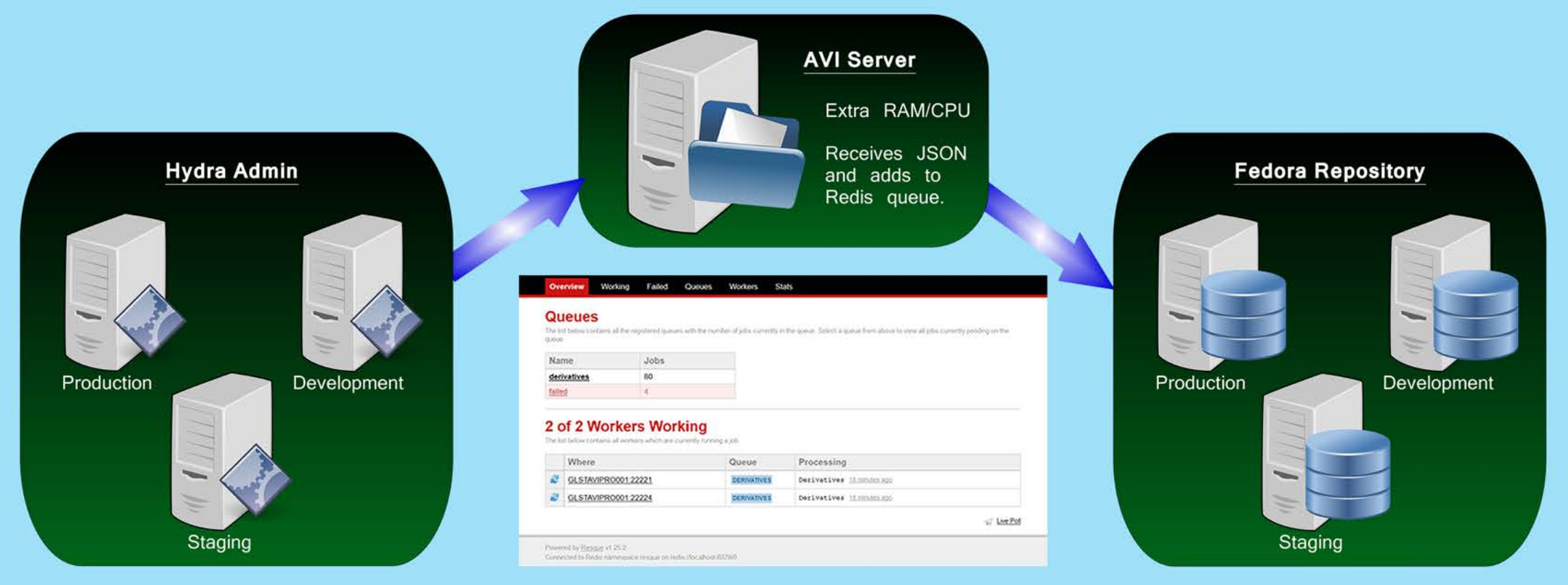
Unlike the average institutional repository use case, our administrative users (libraries) overwhelmingly need to upload objects and metadata in large batches rather than as single items. To this end, we've developed batch ingest functionality that supports uploading items from an Excel spreadsheet.



To provide the flexibility to support metadata from a wide variety of institutions, the spreadsheet template supports over 120 column types corresponding to MODS elements that can be mixed and matched according to the needs of the institution and the collection.

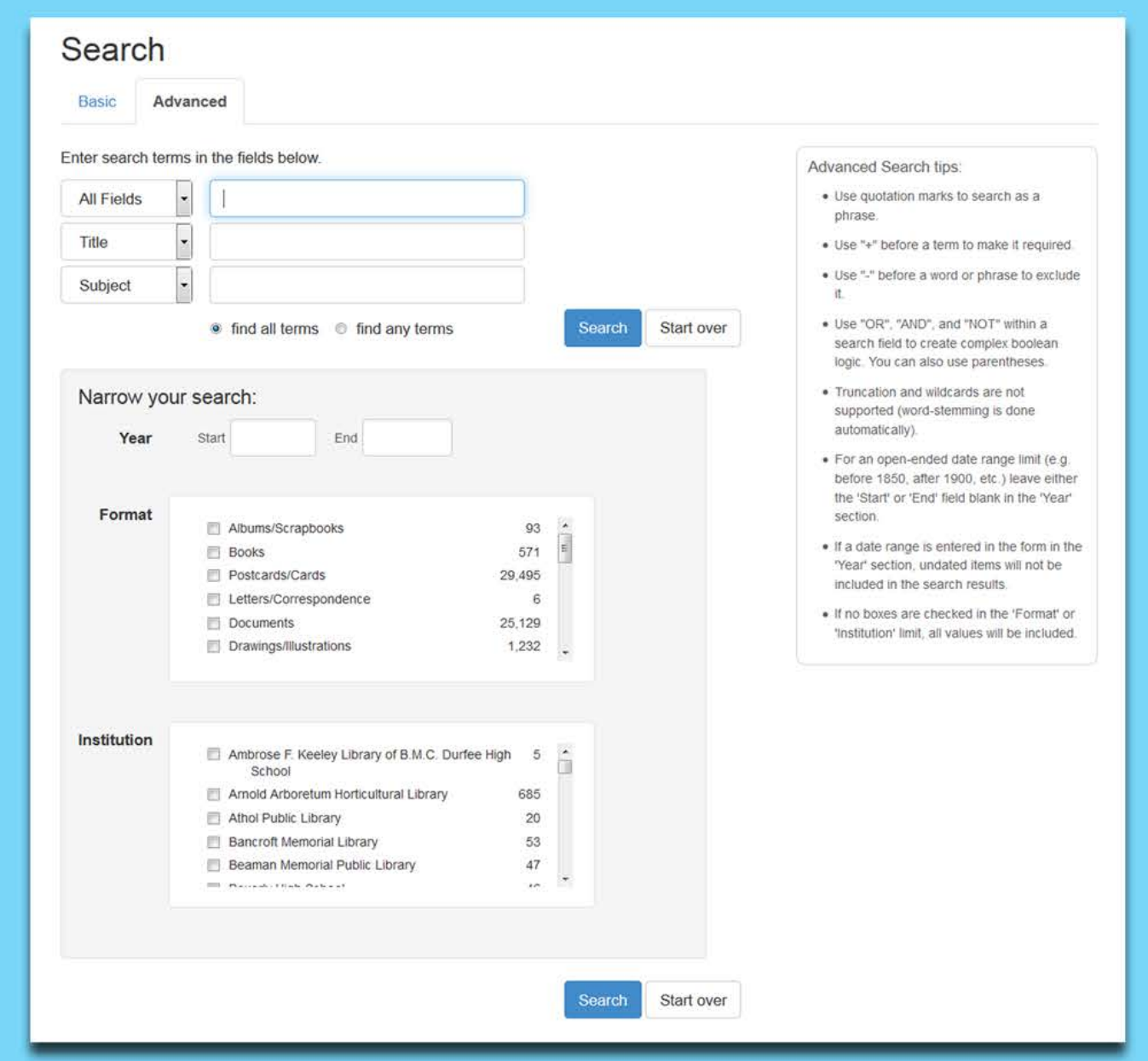
Derivative Creation

Creating derivative files eats up system resources, and it's best to avoid taking RAM away from the public web app while processing GBs of images or video. To deal with this issue we've created an 'avi_processor' Hydra app to handle derivative file creation. During ingest, the admin app sends a request to the 'avi_processor' app (running on its own VM), which initializes a job to create the derivatives and FITS file characterization. The processed files are then written to Fedora upon completion. Processing states are tracked in Solr, so views can be customized accordingly.



Advanced Search

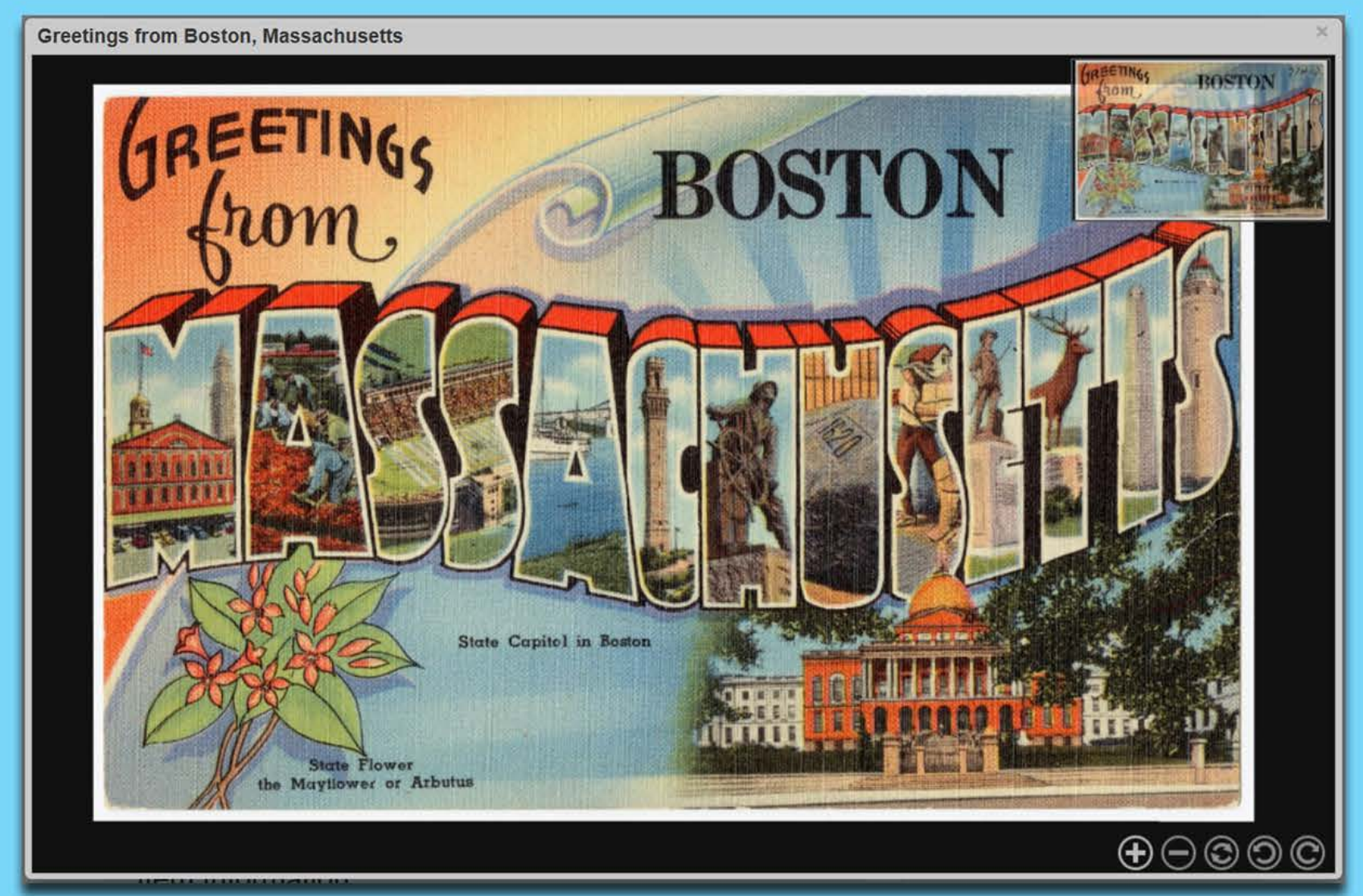
This view expands on the `blacklight_advanced_search` gem, with a date range input and search index selector for each text input field.



The layout has also been modified to resemble common database search interfaces, such as those from EBSCO or ProQuest.

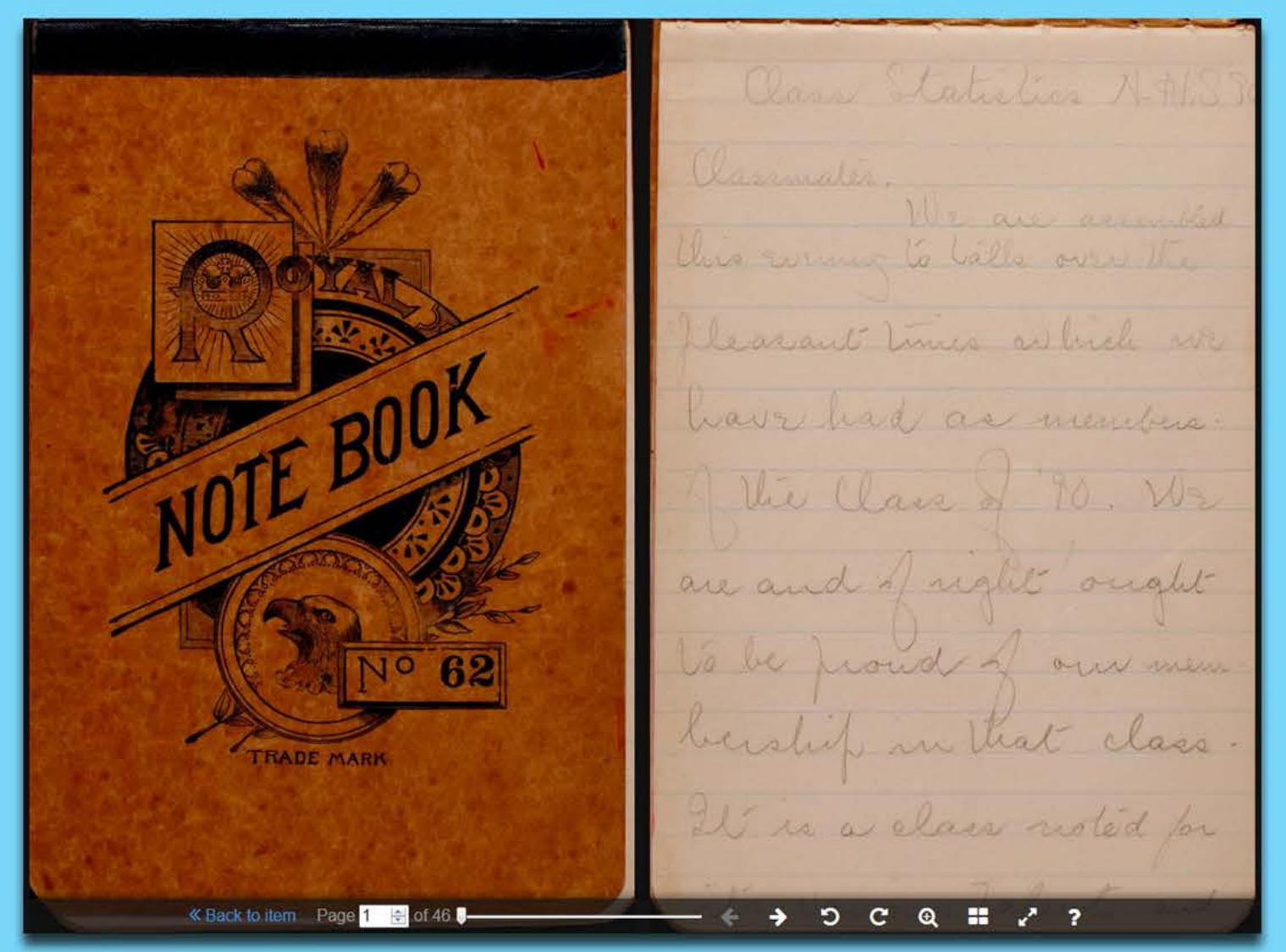
Djatoka + IIIF + Openseadragon

In order to make the Djatoka image server work with the International Image Interoperability Framework, we created a controller that parses IIIF-syntax HTTP requests and converts them to the OpenURL syntax used by Djatoka. The IIIF requests can then be used by the Openseadragon JS library to create feature-rich image viewing functionality.


Greetings from Boston, Massachusetts
GREETINGS from BOSTON MASSACHUSETTS
State Capitol in Boston

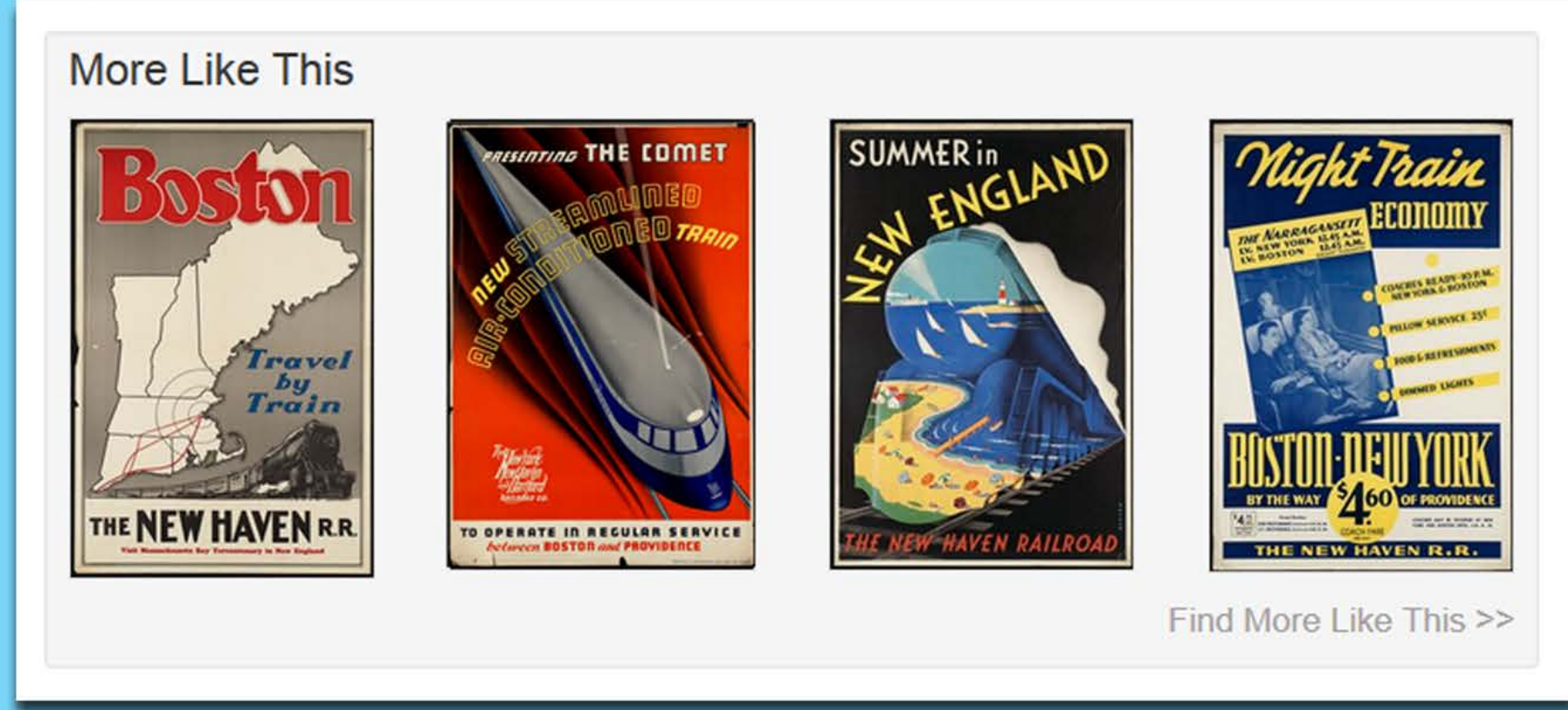
Book Viewer

Based on the WDL-Viewer JS library from the Library of Congress, the integrated book viewer application delivers excellent mobile/tablet responsiveness and a robust feature set: deep zoom using Openseadragon, sequential-page or specific-page navigation, thumbnail browsing, image rotation, full-screen view, and optional mouse-free controls.



More Like This

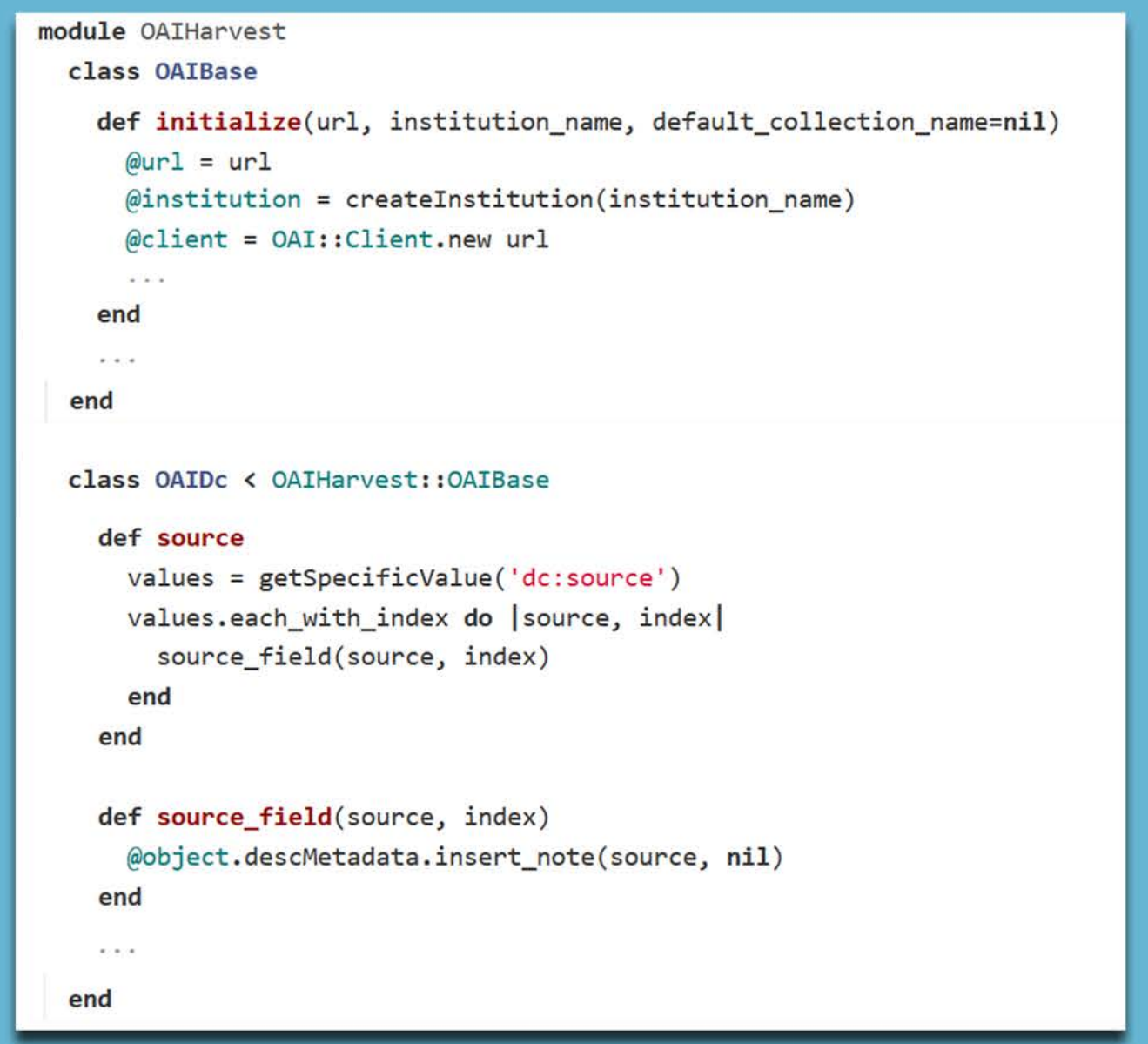
When visitors come from Google or a referring site such as DPLA, they're often going straight to the catalog#show view for an item. In these situations, referring users to similar items becomes a key piece of the UX.



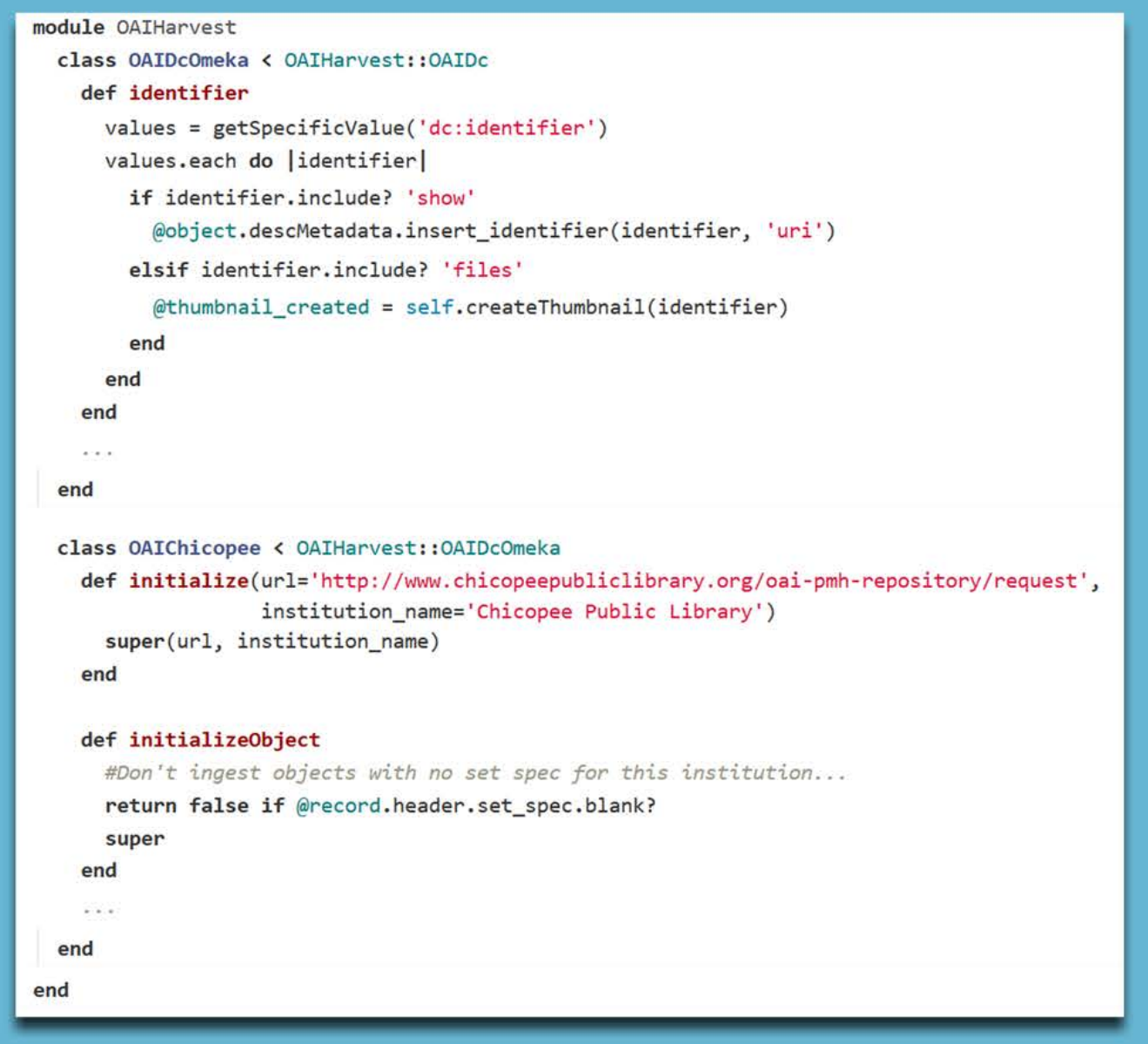
Blacklight provides a module `Blacklight::Solr::Document::MoreLikeThis` that uses Solr's MoreLikeThisComponent within the default SearchHandler to provide a configurable set of similar results based on fields defined in solrconfig.xml. Once this is set up, the possibilities for displaying related items in the view are wide open.

OAI Harvest

Taking the `oai` gem as a starting point, we've created functionality to harvest metadata records from an OAI-PMH feed and store records and object thumbnails as objects in Fedora. The code uses an object-oriented approach to dealing with the characteristics and idiosyncrasies of common OAI transmission schemas and provider systems such as CONTENTdm and Omeka. To date, we've harvested over 44K records from 10 different OAI end-points.



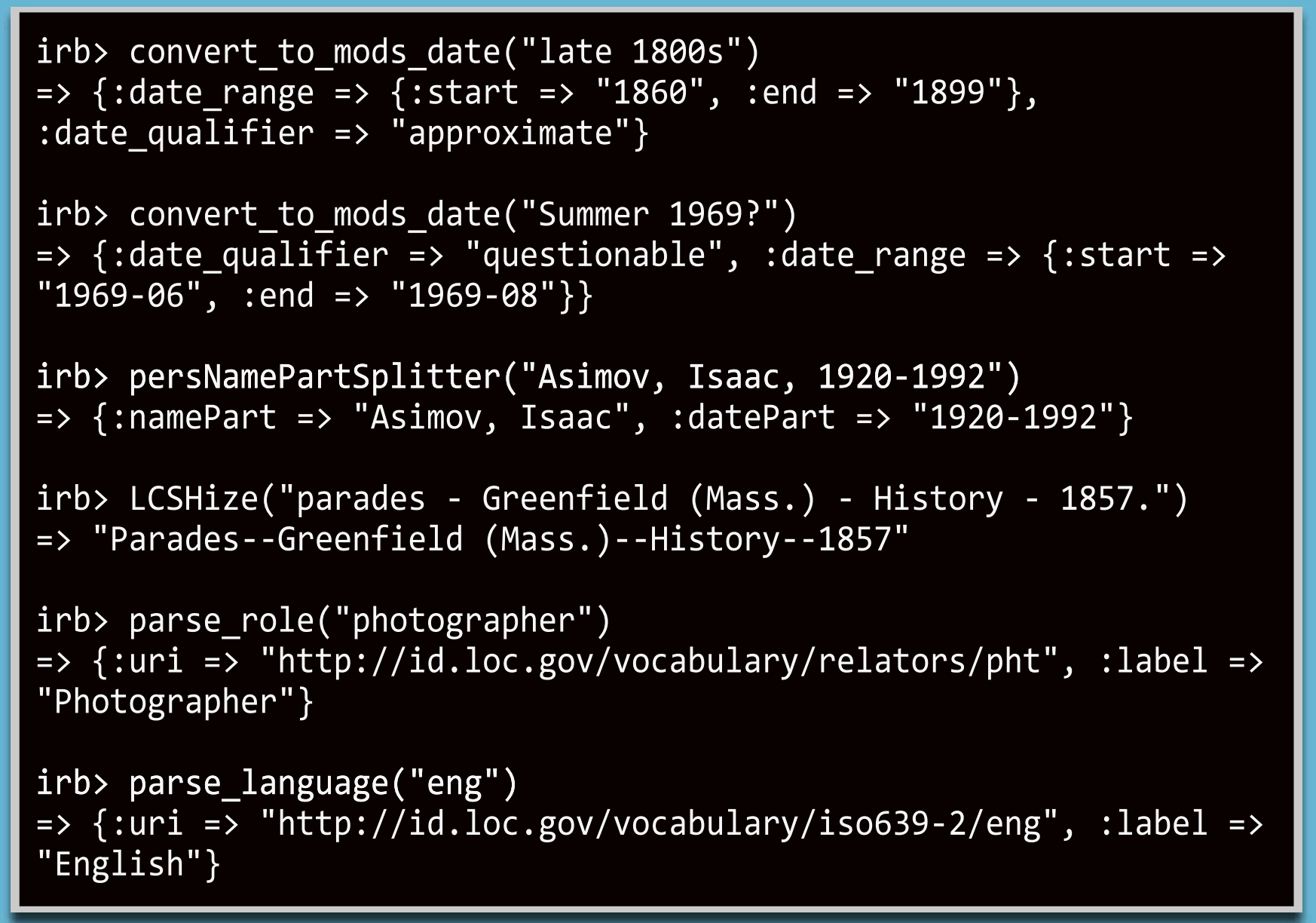
```ruby
module OAIHarvest
  class OAIBase
    def initialize(url, institution_name, default_collection_name=nil)
      @url = url
      @institution = createInstitution(institution_name)
      @client = OAI::Client.new url
      ...
    end
    ...
  end

  class OAIDc < OAIHarvest::OAIBase
    def source
      values = getSpecificValue('dc:source')
      values.each_with_index do |source, index|
        source_field(source, index)
      end
    end

    def source_field(source, index)
      @object.descMetadata.insert_note(source, nil)
    end
    ...
  end
```



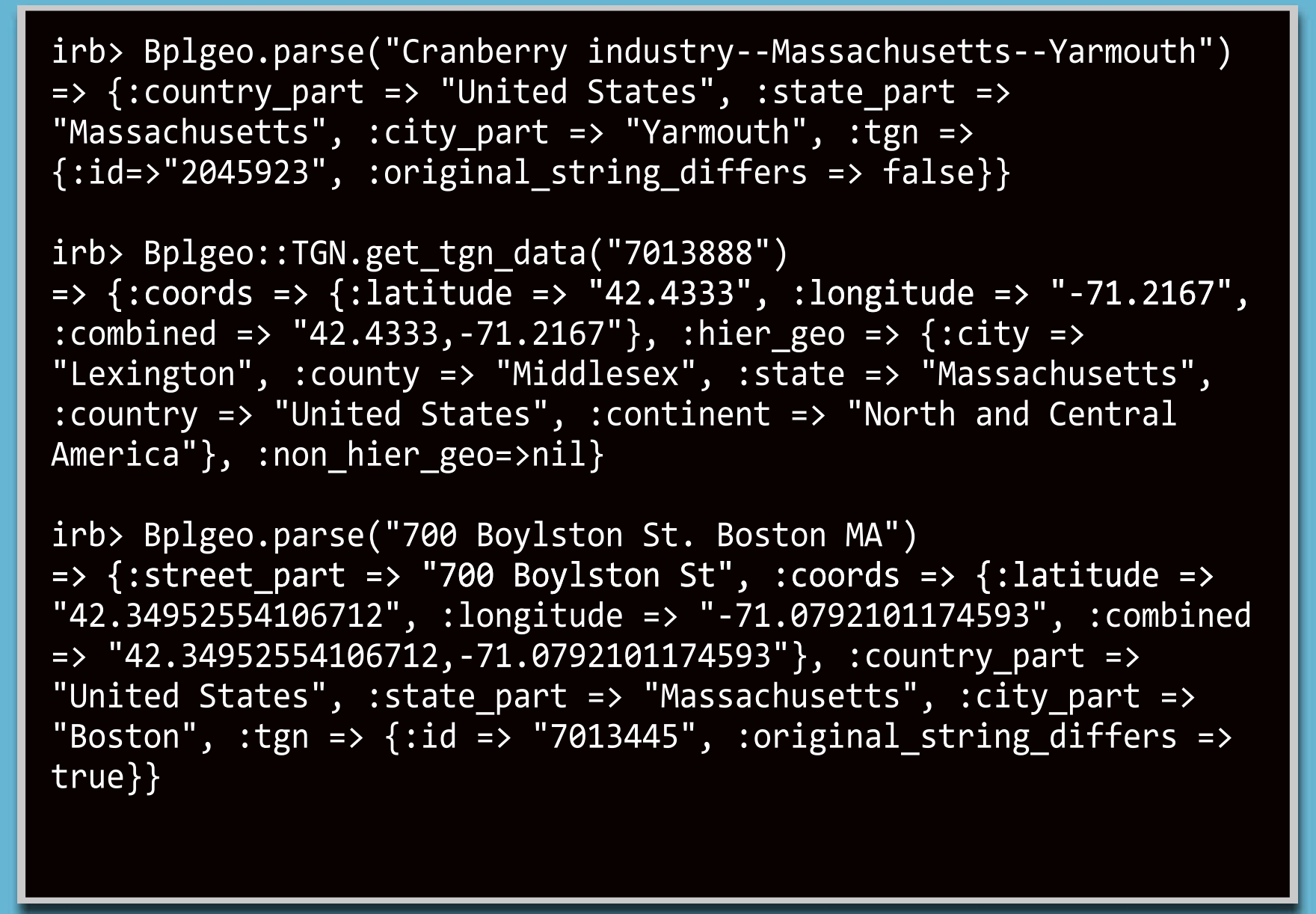
```ruby
module OAIHarvest
  class OAIDcOmeka < OAIHarvest::OAIDc
    def identifier
      values = getSpecificValue('dc:identifier')
      values.each do |identifier|
        if identifier.include? 'show'
          @object.descMetadata.insert_identifier(identifier, 'uri')
        elsif identifier.include? 'files'
          @thumbnail_created = self.createThumbnail(identifier)
        end
      end
    end
    ...
  end

  class OAIChicopee < OAIHarvest::OAIDcOmeka
    def initialize(url='http://www.chicopeepubliclibrary.org/oai-pmh-repository/request',
                   institution_name='Chicopee Public Library')
      super(url, institution_name)
    end

    def initializeObject
      #Don't ingest objects with no set spec for this institution...
      return false if @record.header.set_spec.blank?
      super
    end
    ...
  end
end
```

Authority Control

Never trust ~~user~~ library-submitted data! What happens when you dump together metadata from dozens of sources and schemas? Chaos, that's what. To combat the cacophony, we use a number of cleanup routines based on the `questioning_authority` and `bplgeo` gems that turn messy, inconsistent data into standardized headings using authorities from the Library of Congress and the Getty Thesaurus of Geographic Names.



```
irb> convert_to_mods_date("late 1800s")
=> {:date_range => {:start => "1860", :end => "1899"},
:date_qualifier => "approximate"}

irb> convert_to_mods_date("Summer 1969?")
=> {:date_qualifier => "questionable", :date_range => {:start =>
"1969-06", :end => "1969-08"}}

irb> persNamePartSplitter("Asimov, Isaac, 1920-1992")
=> {:namePart => "Asimov, Isaac", :datePart => "1920-1992"}

irb> LCSHize("parades - Greenfield (Mass.) - History - 1857.")
=> "Parades--Greenfield (Mass.)--History--1857"

irb> parse_role("photographer")
=> {:uri => "http://id.loc.gov/vocabulary/relators/pht", :label =>
"Photographer"}

irb> parse_language("eng")
=> {:uri => "http://id.loc.gov/vocabulary/iso639-2/eng", :label =>
"English"}
```



```
irb> Bplgeo.parse("Cranberry industry--Massachusetts--Yarmouth")
=> {:country_part => "United States", :state_part =>
"Massachusetts", :city_part => "Yarmouth", :tgn =>
{:id=>"2045923", :original_string_differs => false}}

irb> Bplgeo::TGN.get_tgn_data("7013888")
=> {:coords => {:latitude => "42.4333", :longitude => "-71.2167",
:combined => "42.4333,-71.2167"}, :hier_geo => {:city =>
"Lexington", :county => "Middlesex", :state => "Massachusetts",
:country => "United States", :continent => "North and Central
America"}, :non_hier_geo=>nil}

irb> Bplgeo.parse("700 Boylston St. Boston MA")
=> {:street_part => "700 Boylston St", :coords => {:latitude =>
"42.34952554106712", :longitude => "-71.0792101174593", :combined
=> "42.34952554106712,-71.0792101174593"}, :country_part =>
"United States", :state_part => "Massachusetts", :city_part =>
"Boston", :tgn => {:id => "7013445", :original_string_differs =>
true}}
```

Steven Anderson sanderson@bpl.org | Eben English eenglish@bpl.org | Boston Public Library

Digital Commonwealth