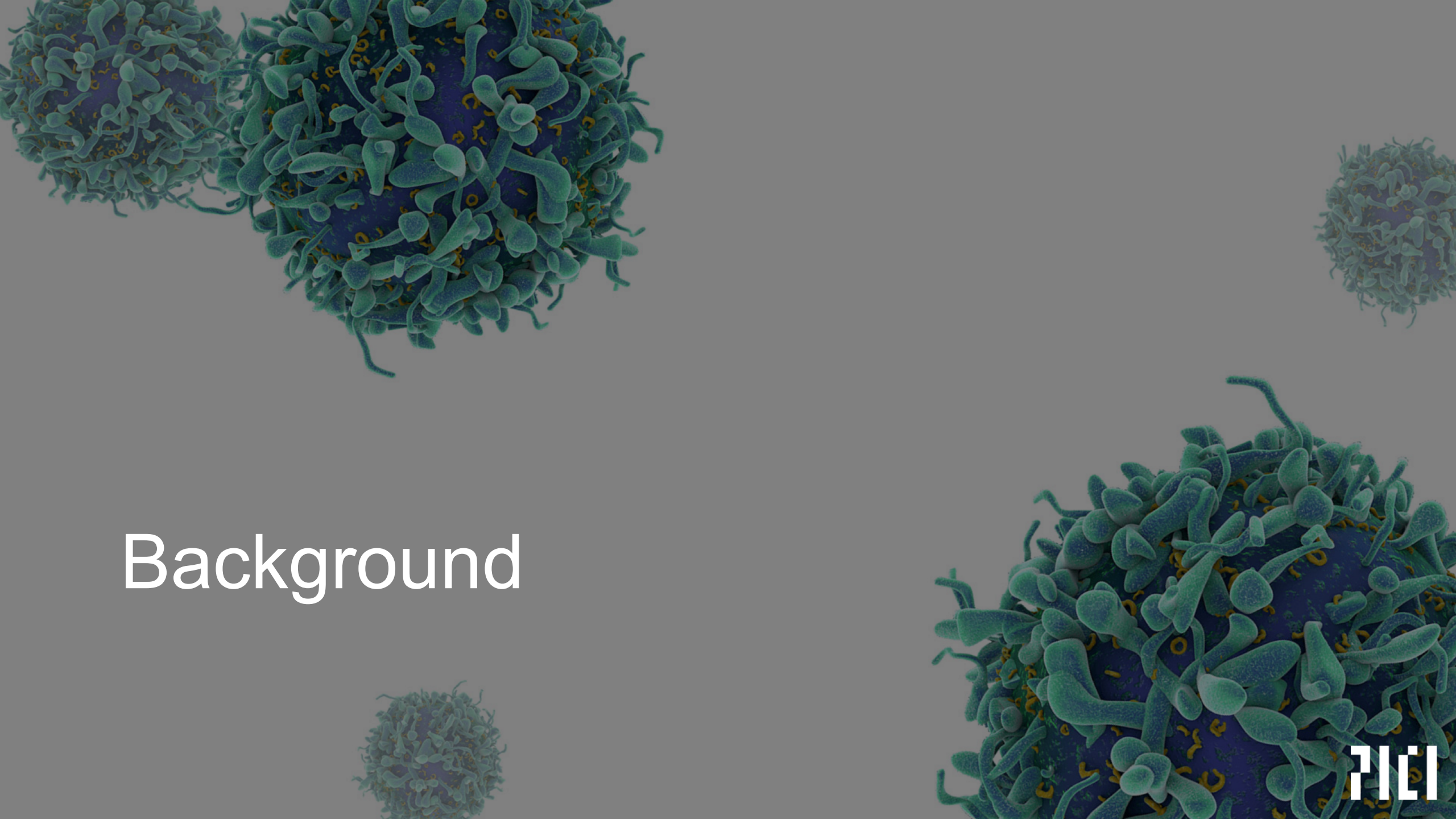
A visual query builder for knowledge graphs

Mike Travers
mtravers@parkerici.org

PARKER INSTITUTE FOR CANCER IMMUNOTHERAPY



Background

Parker Institute for Cancer Immunotherapy

Mission

To accelerate the development of *breakthrough immune therapies* to turn cancer into a *curable disease*.

Cancer Immunotherapy

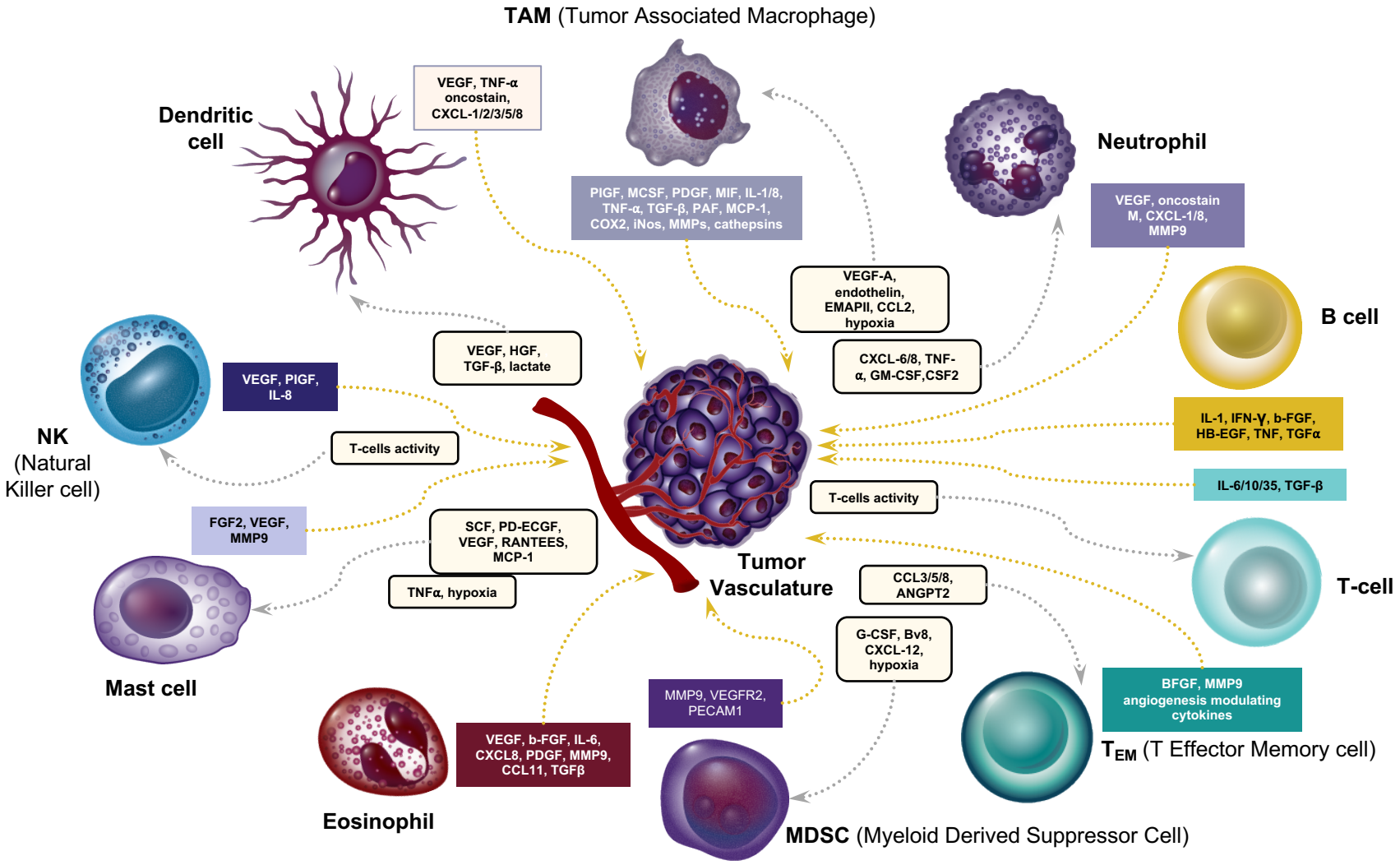
Treatments that use the body's immune system to destroy cancerous cells.

Benefits

- Works on non-local tumors and many types of cancer
- Lower toxicity than chemo
- Works well in conjunction with other therapies
- In some patients, **long-lasting responses.**

Relatively new area, lots of unknowns, lots of research to be done

Tumor–Immune interactions are complex, but critical to understand



1. Stockmann C et al. Front Oncol. 2014;4:69.
2. Balkwill FR et al. J Cell Sci. 2012;125(Pt 23):5591-5596.

PICI Informatics Team

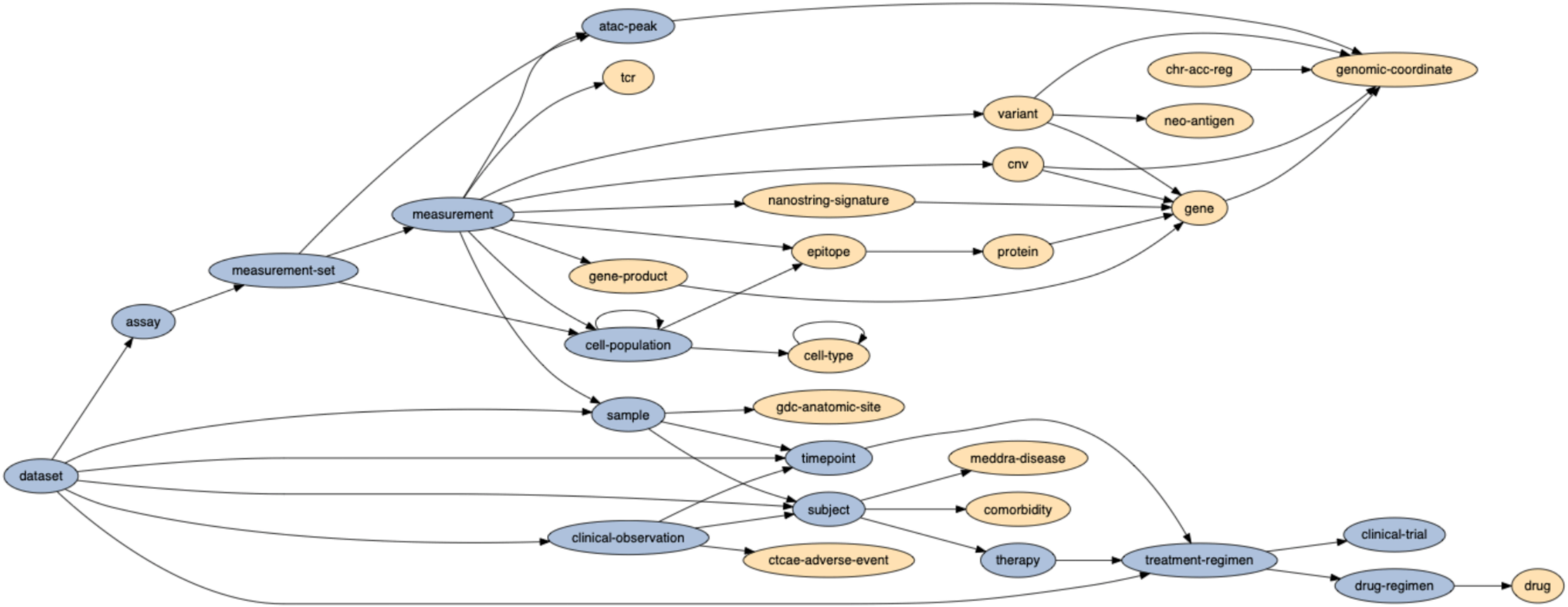
Contact us: **data@parkerici.org**

CANDEL: CANcer Data & Evidence Library

- Knowledgebase for research data
- Based on Datomic (and built in partnership with Cognitect)
- ingesting **experimental** and **reference** data from our own trials and from published research
- organizing it into a common framework
- providing it to downstream tasks (querying, analysis, visualization)
- Some objects: subjects, samples, genes, variants, clinical trials

StrangeLoop 2019: **Building a Unified Cancer Immunotherapy Data Library**
Lacey Kitsch, Ben Kamphaus
https://youtu.be/vwZxHVcfwuw

Medium-complex schema



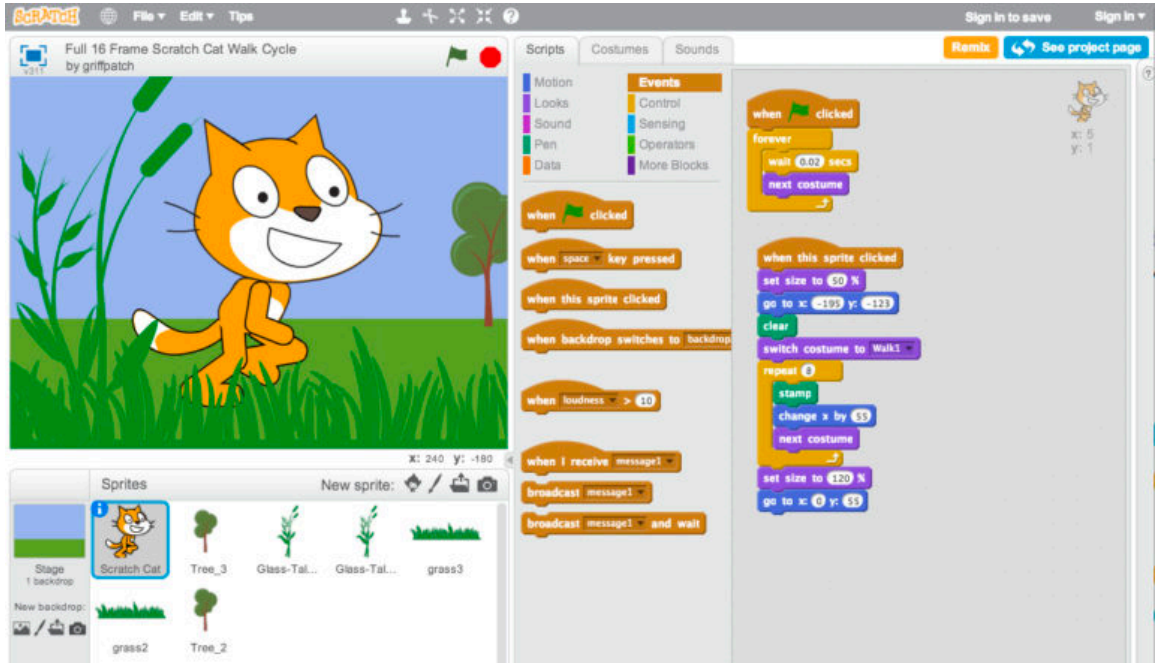
Block languages

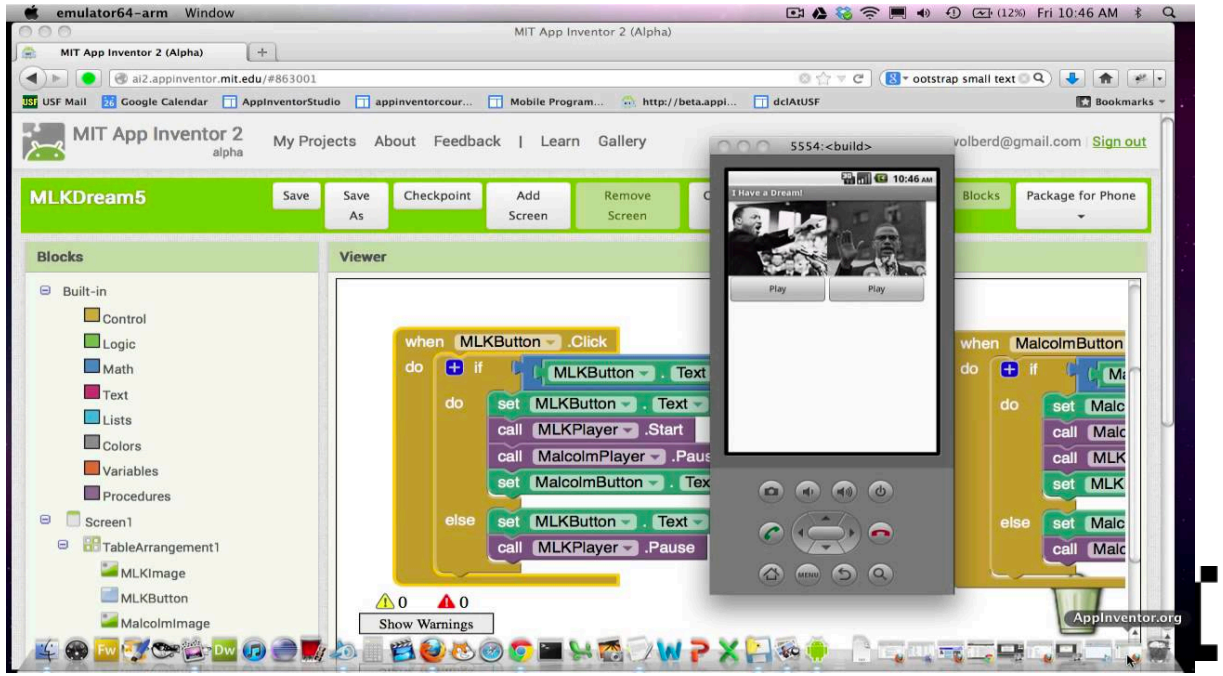
Scratch

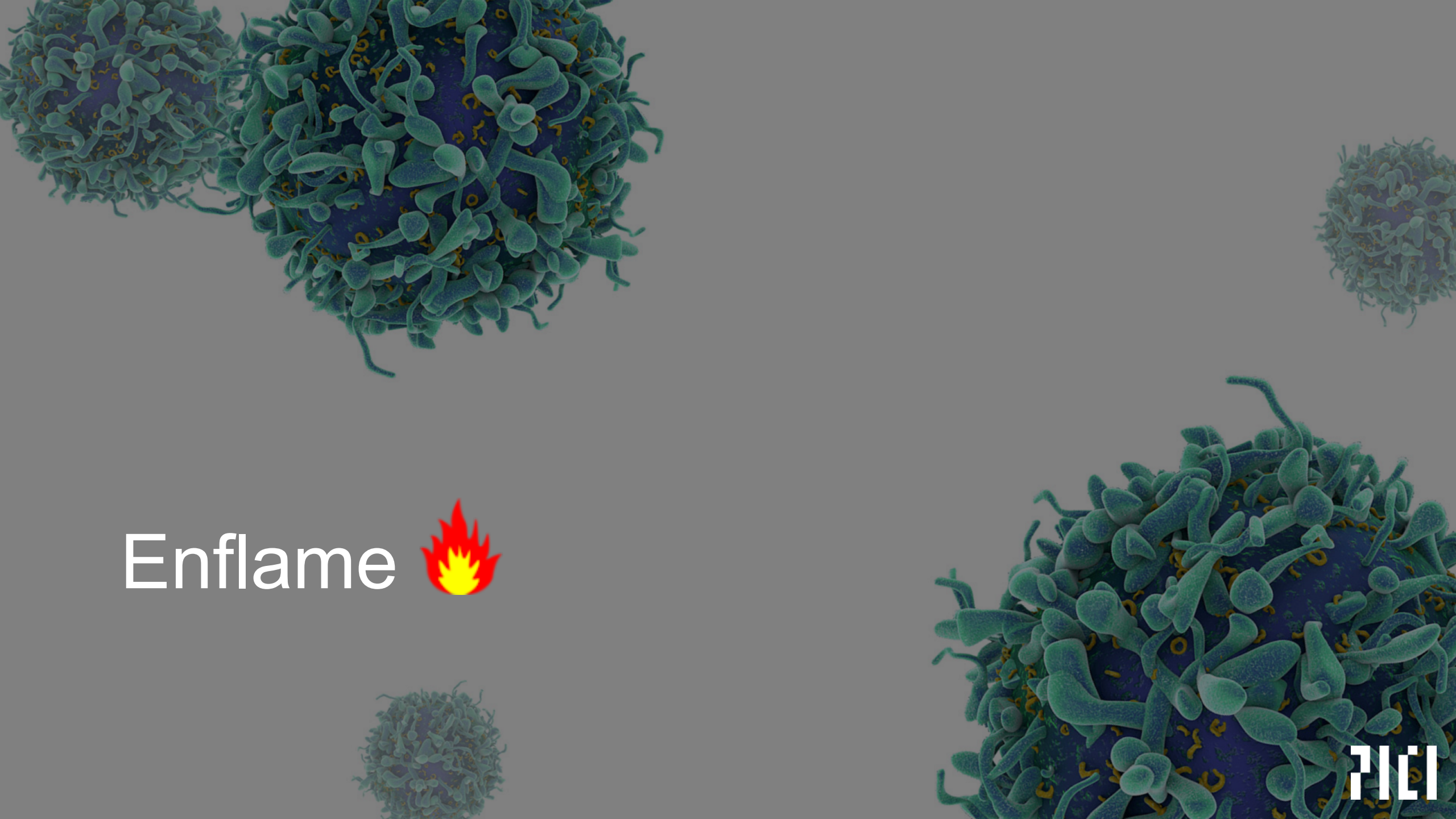
MIT Media Lab (2003).

Scratch-like languages:

- **Tynker** (commercial)
- **Snap** from Berkeley (high-order procedures!)
- Android app inventor / **Blockly** (from Google)






Enflame

DB dannys-database-1.2

Query Go

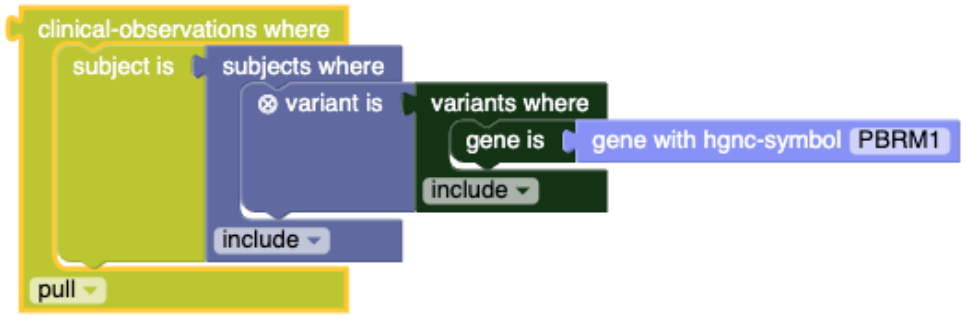
```
{:find
 ((pull
   ?clinical-observation1
   [*
    {:clinical-observation/ctcae-adve
     [:db/id :ctcae-adverse-event/nam
    {:clinical-observation/timepoint
    {:clinical-observation/subject [:
  (pull ?subject1 [:db/id :subject/id
  (pull ?variant1 [:db/id :variant/id
 :where
 ([?measurement1 :measurement/variant
  [?measurement1 :measurement/sample
  [?sample1 :sample/subject ?subject1
  [?gene1 :gene/hgnc-symbol "PBRM1"]
  [?variant1 :variant/gene ?gene1]
  [?clinical-observation1 :clinical-o
```

Wick

Share Library

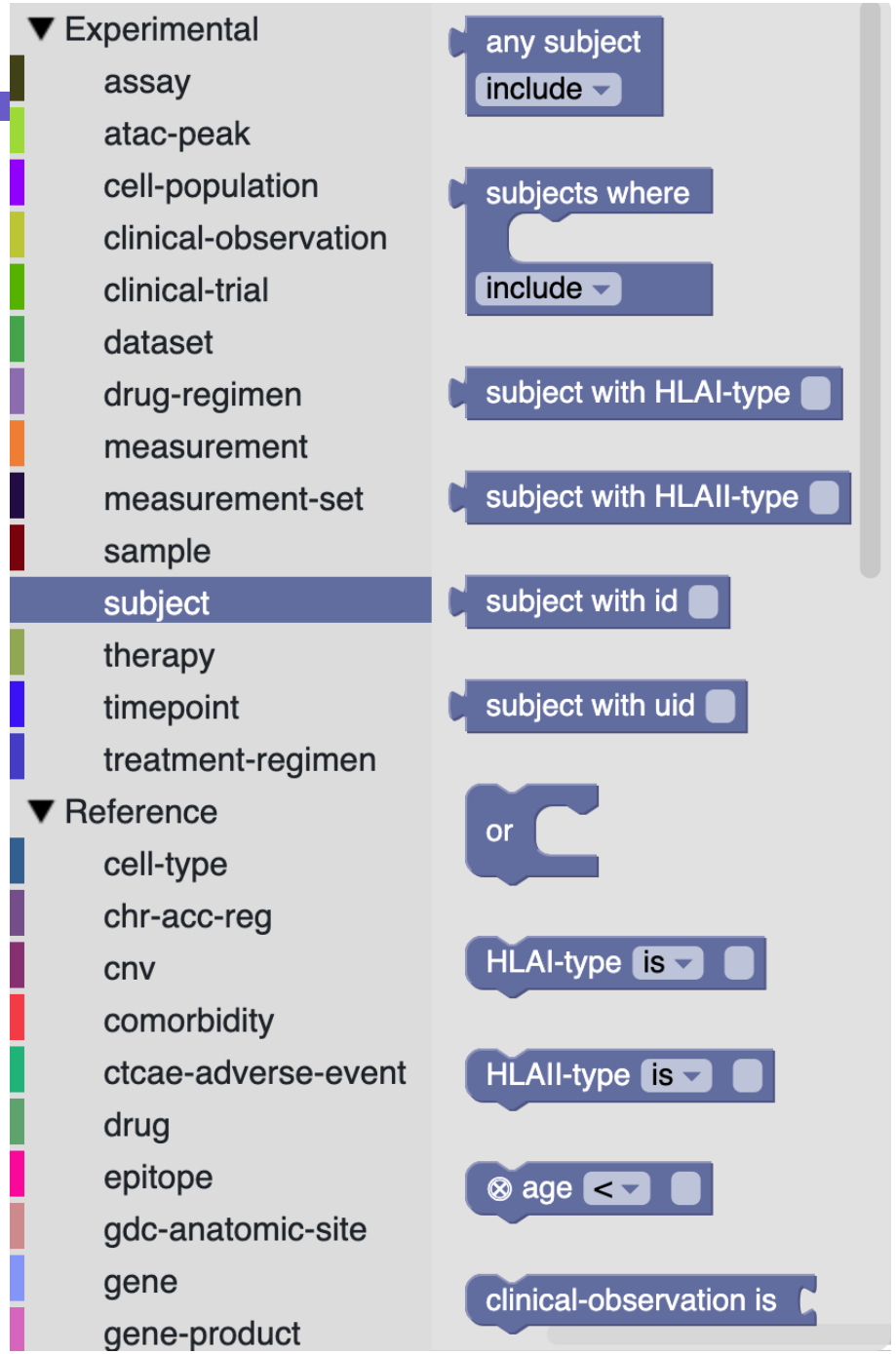
Browser

- Experimental
  - assay
  - atac-peak
  - cell-population
  - clinical-observation
  - clinical-trial
  - dataset
  - drug-regimen
  - measurement
  - measurement-set
  - sample
  - subject
  - therapy
  - timepoint
  - treatment-regimen
- Reference
  - cell-type
  - chr-acc-reg
  - cnv
  - comorbidity
  - ctcae-adverse-event
  - drug
  - epitope
  - gdc-anatomic-site
  - gene
  - gene-product



Showing 6 of 6 Download

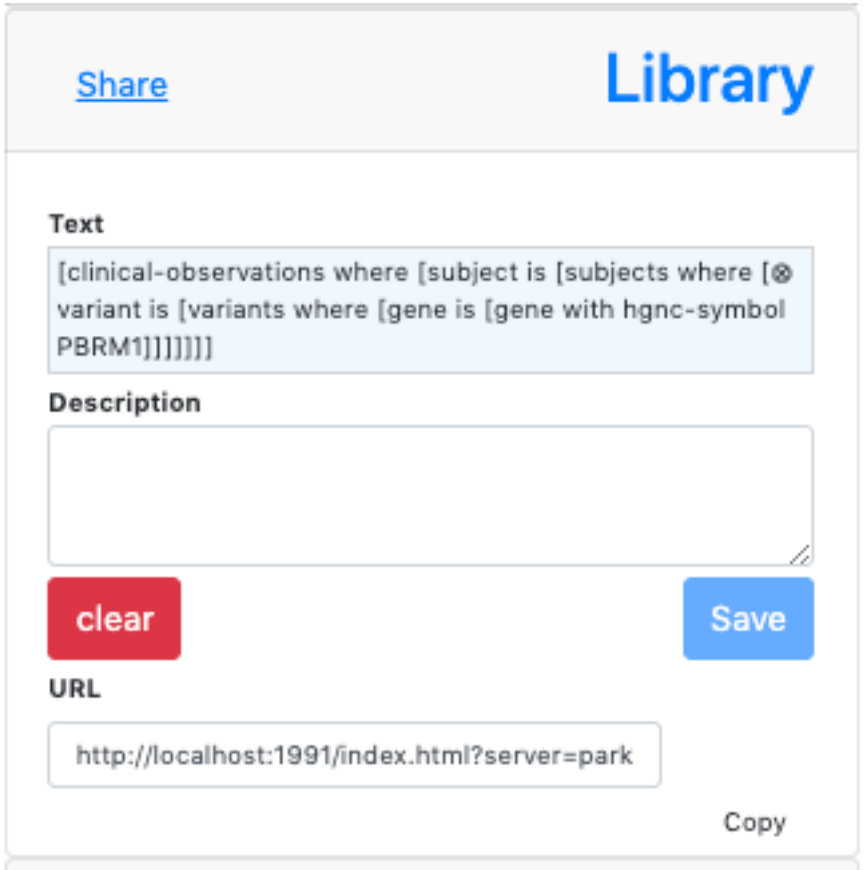
clinical-observation	age	recist	responder	subject	timepoint	subject	variant
17592186367292		PR		M4945	Pembrolizumab-2wk/eos	M4945	GRCh38:chr3:+:52696148:52696148/C/A
17592186367289	66			M4945	Pembrolizumab-2wk/baseline	M4945	GRCh38:chr3:+:52696148:52696148/C/A
17592186367298		PD		MA7027	Pembrolizumab-2wk/eos	MA7027	GRCh38:chr3:+:52598231:52598231/G/A
17592186367296	56			MA7027	Pembrolizumab-2wk/baseline	MA7027	GRCh38:chr3:+:52598231:52598231/G/A




```
{:find ((pull ?subject1 [:db/id :subject/id]))
 :where ([?subject1 :subject/race :race/asian])}
```

Side panes

DB selection; query in various formats, other controls



Design philosophy

A slightly higher-level language than Datalog

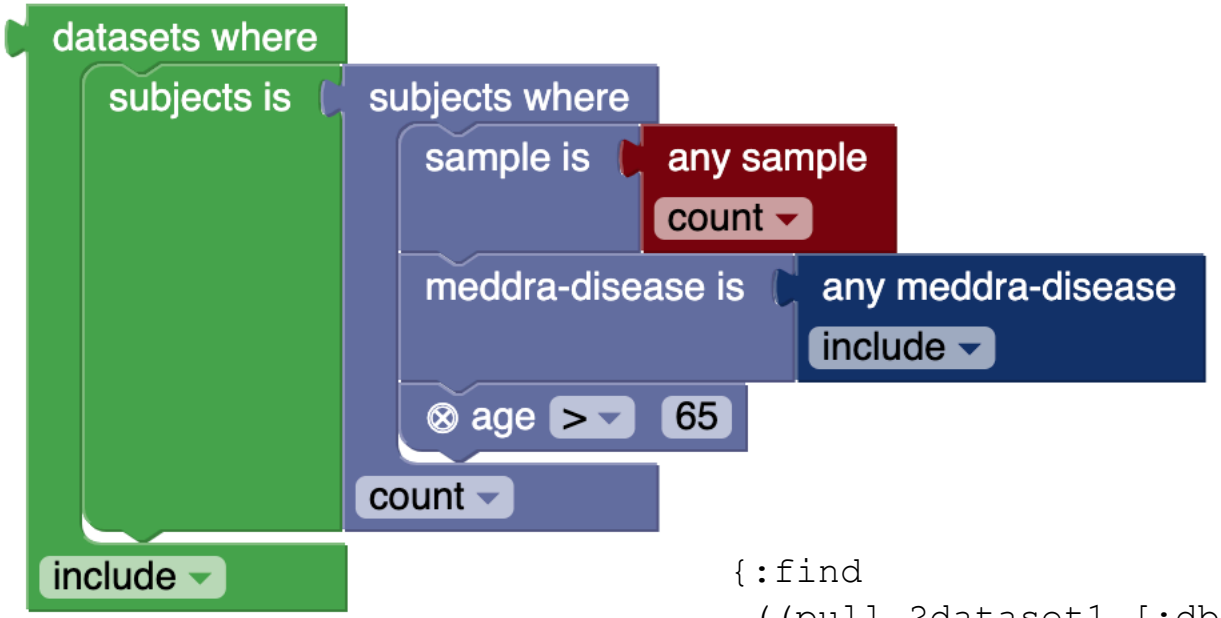
Training wheels:

shows how to do it the adult way

doesn't need to provide full power of underlying system

```
{:find ((pull ?gene1 [:db/id :gene/hgnc-symbol]) ?hgnc-symbol1),
 :where
 ([(re-find ?regex ?hgnc-symbol1)]
  [(re-pattern ".*XX.*") ?regex]
  [?gene1 :gene/hgnc-symbol ?hgnc-symbol1])}
```

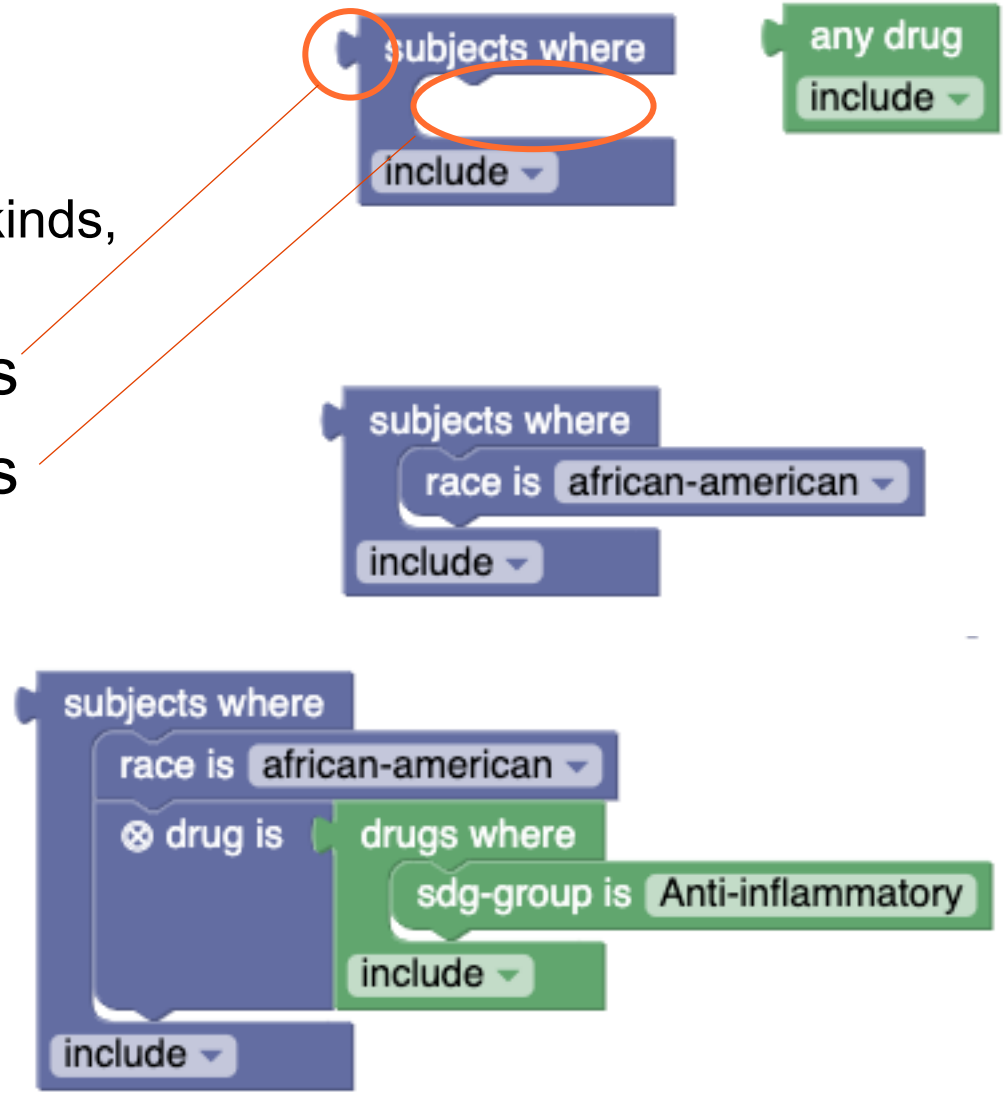
A more complex example




```
{:find
 ((pull ?dataset1 [:db/id :dataset/name])
  (count-distinct ?subject1)
  (count-distinct ?sample1)
  (pull ?meddra-disease1 [:db/id :meddra-disease/preferred-name]) ?age1),
 :where
 ([?clinical-observation1 :clinical-observation/subject ?subject1]
  [(> ?age1 65)]
  [?clinical-observation1 :clinical-observation/age ?age1]
  [?subject1 :subject/meddra-disease ?meddra-disease1]
  [?sample1 :sample/subject ?subject1]
  [?sample1 :sample/id ?id1]
  [?dataset1 :dataset/subjects ?subject1])}
```

Design details

- Class (kind) mapped to color
- works OK because there are only ~30 kinds, won't scale
- Output nub produces sets of objects
- Statement input used for constraints (where clauses) because it's n-ary
- Constraint blocks primitives
- Constraint block subqueries



Implementation

Components

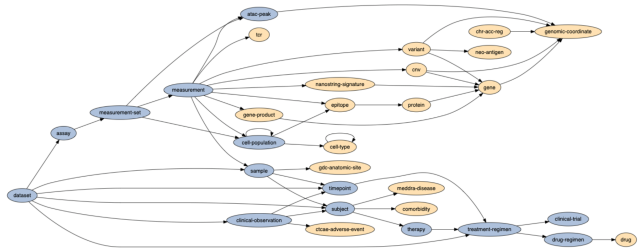
Underlying technologies

- **Blockly**
  open source Google library for block UI
- **Clojure[script]**
- **Re-frame**
- **Alzabo**
  schema representation

Interacts with

- **CANDEL**
  Datomic database
- **Gaslight**
  query server
- **Wick**
  R Datalog package

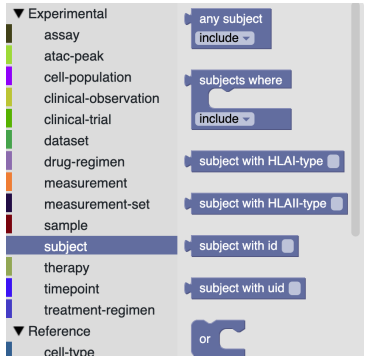
Flows

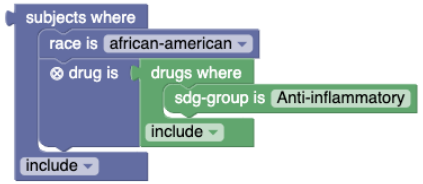


CANDEL schema

Alzabo schema

Toolbox definition





Blockly XML

Compact form

Query Generation

Gaslight / Datomic

Result display

Output specifier

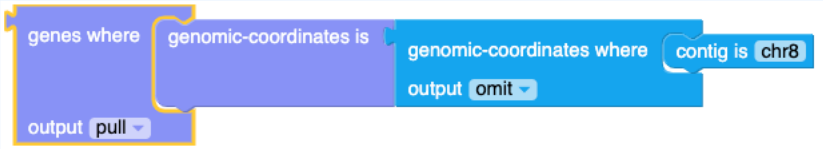
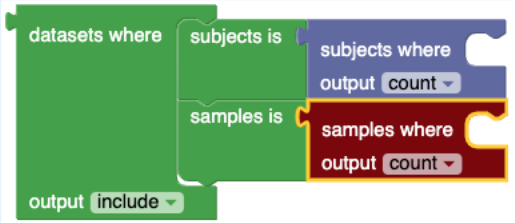
A **query entity block** produces a set of entities of a given type. For instance, this produces the set of all subjects:

Query entity blocks have an additional selector that lets you specify the output type. The options are:

- `include`: (default) include the entity itself and its label (unique-id) if available
- `pull`: include the entity and all of its attributes
- `count`: don't return the entity itself, but instead the count of its unique values based on the rest of the query
- `omit`: don't return anything for this entity

Sharing Features: Library

Enflame: Library

[subjects where [⊗ drug is [drugs]]]	Drugs with subject count	
[samples where [tumor-type is *]]	Sample count by tumor type	
[samples where [gdc-anatomic-site is [gdc-anatomic-sites]]]	Sample counts by anatomical region	
[drug with preferred-name]		
[genes where [genomic-coordinates is [genomic-coordinates where [contig is chr8]]]]	Genes on a given chromosome	
[subjects where [⊗ treatment-regimen is [treatment-regimen with name aPD1]]]	Subjects with a particular treatment regimen	
[datasets where [subjects is [subjects]] and [samples is [samples]]]	Dataset census	

Alzabo

Schema transformation

(into something basically equivalent to RDFS)

- classes
- attributes
- entity values (eg race, cancer stage)

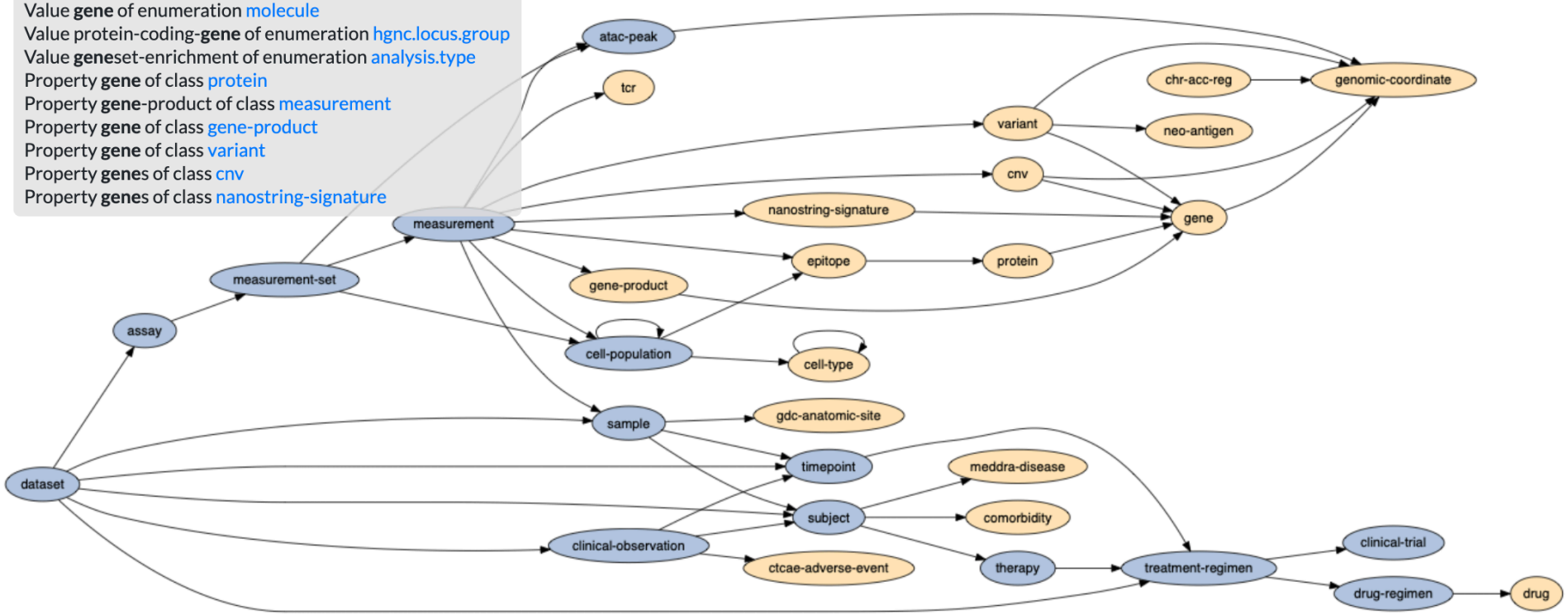
Similar to Hodur https://github.com/hodur-org

Generates documentation (graphviz, html, search widget)

CANDEL Schema

search: gene

Class **gene**
Class **gene**-product
Value **gene** of enumeration molecule
Value protein-coding-**gene** of enumeration hgnc.locus.group
Value **gene**set-enrichment of enumeration analysis.type
Property **gene** of class protein
Property **gene**-product of class measurement
Property **gene** of class gene-product
Property **gene** of class variant
Property **gene**s of class cnv
Property **gene**s of class nanostring-signature



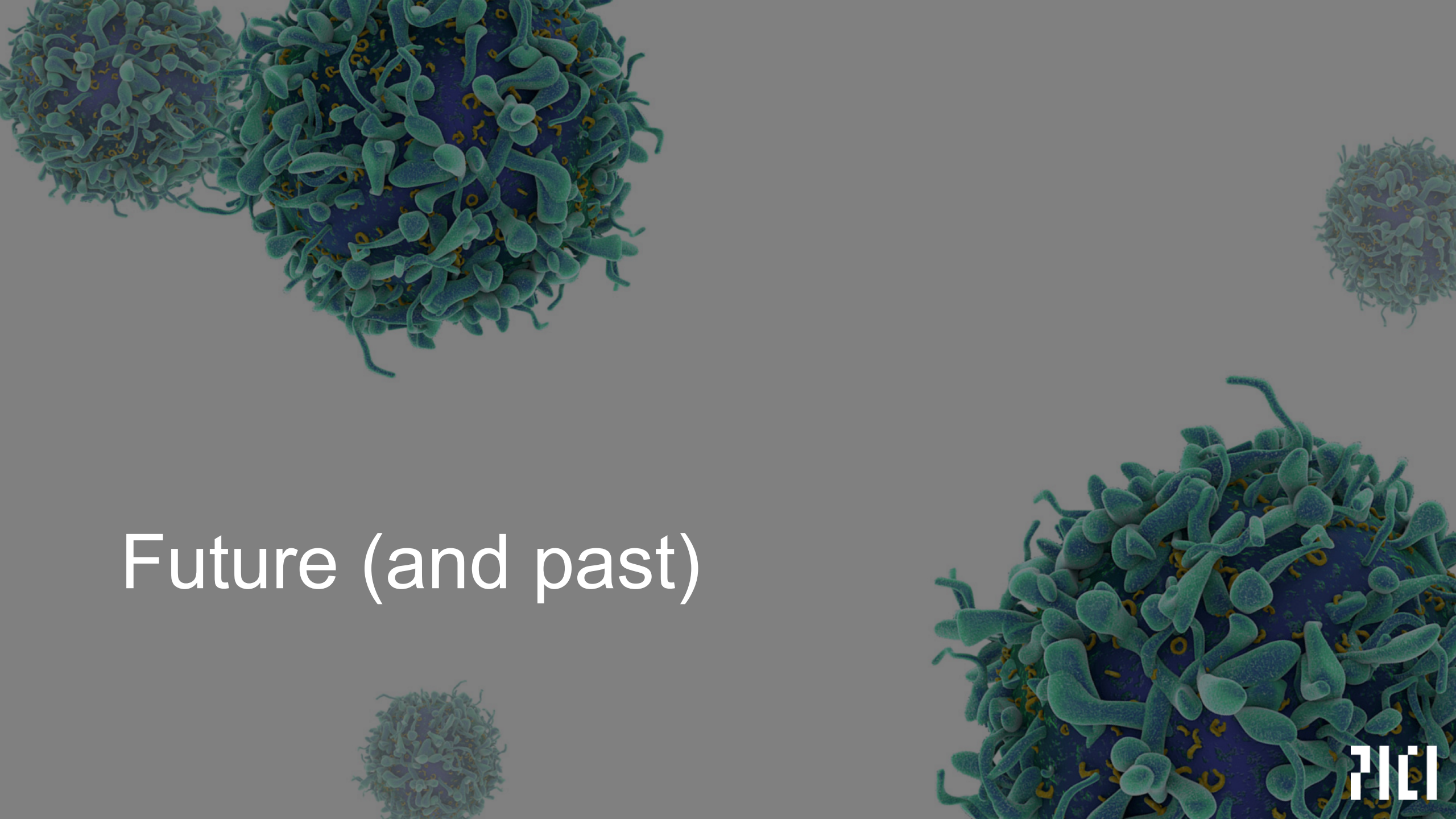
Kinds

reference

- cell-type
- chr-acc-reg
- cnv
- comorbidity
- ctcae-adverse-event

Enums

- ae.grade
- analysis.type
- assembly
- candel
- ethnicity
- hgnc.locus.group
- molecule



Future (and past)

Futures

Usability

- Hide some of schema complexity
- More control over results
- Tighter integration with Alzabo graph
- User testing

Support more complex queries

"How many patients who experienced toxicity on any given therapy had a microbiome sample profiled within 10 days of the toxicity event?"

Adapt to other databases

- Including SPARQL/Linked Data

Open source (someday)

Possible components to open source:

Alzabo

Blockoid (thin CLJS wrapper for Blockly)

Querulous (general graph query generator)

Ultimately would like to make this useful for other datasource, eg SPARQL endpoints like Wikidata.

Related work

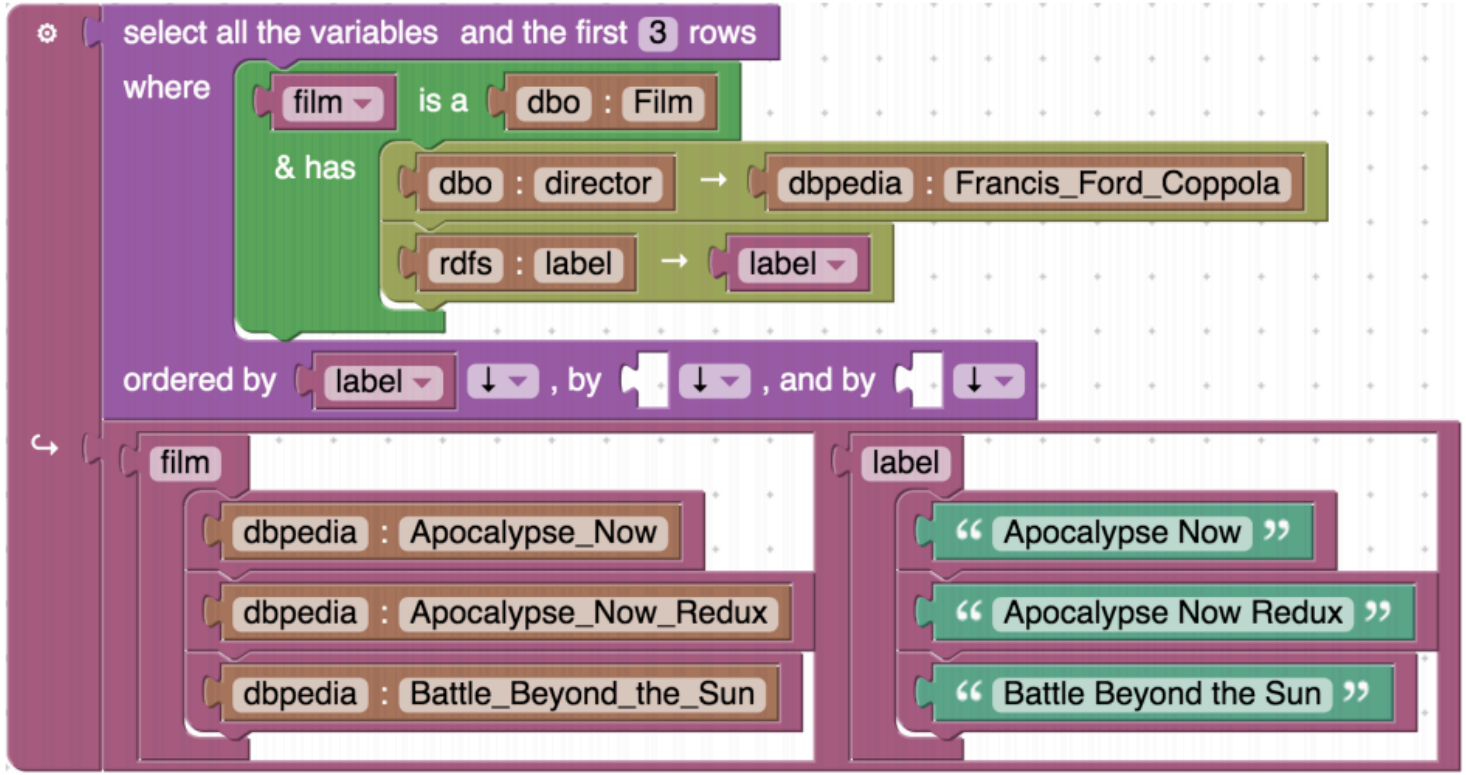
SPARQL Playground: a Block Programming Tool to Experiment with SPARQL

Paolo Bottoni and Miguel Ceriani, 2015

http://sparqlblocks.org/demo/

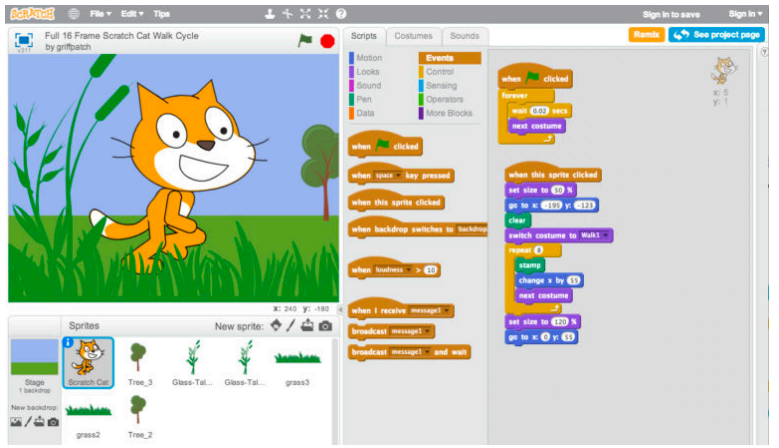
Has a lower-level approach that exposes SPARQL elements directly.

Avoids the semantic type/color problem.

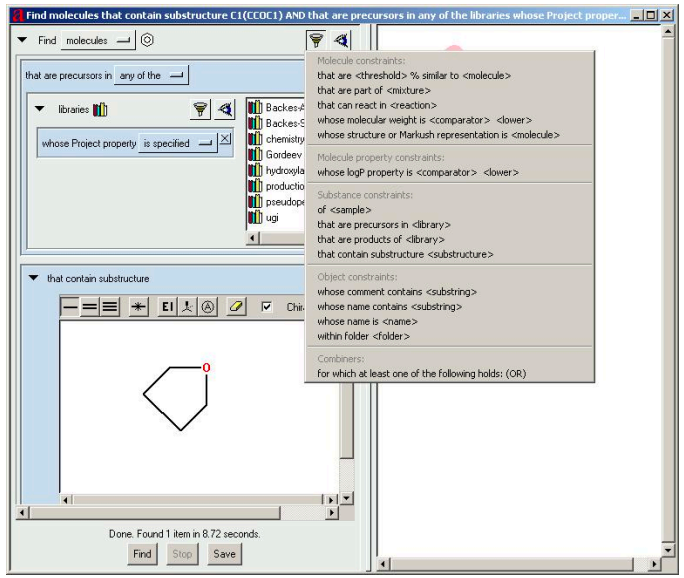


Some history

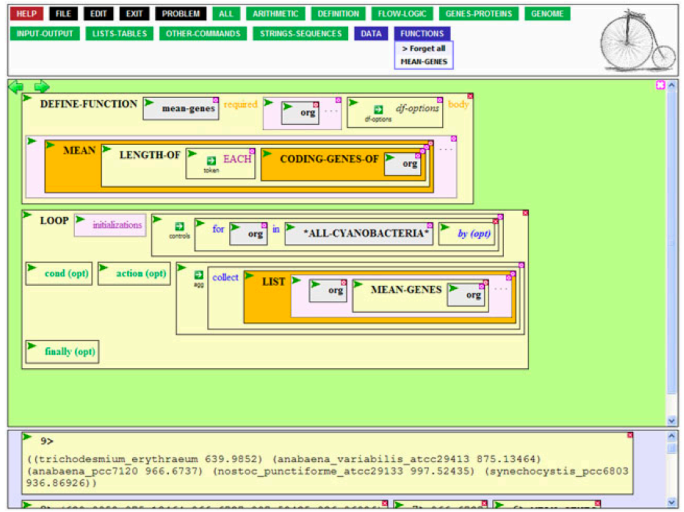
Behave, 1996



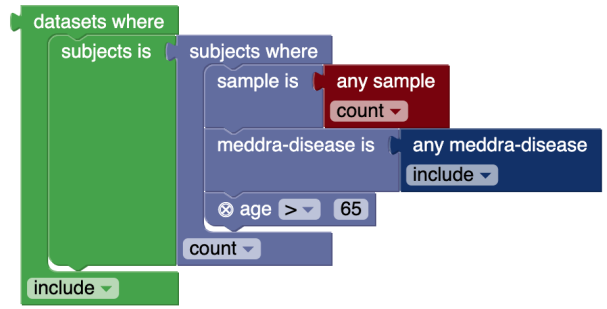
Scratch, 2003



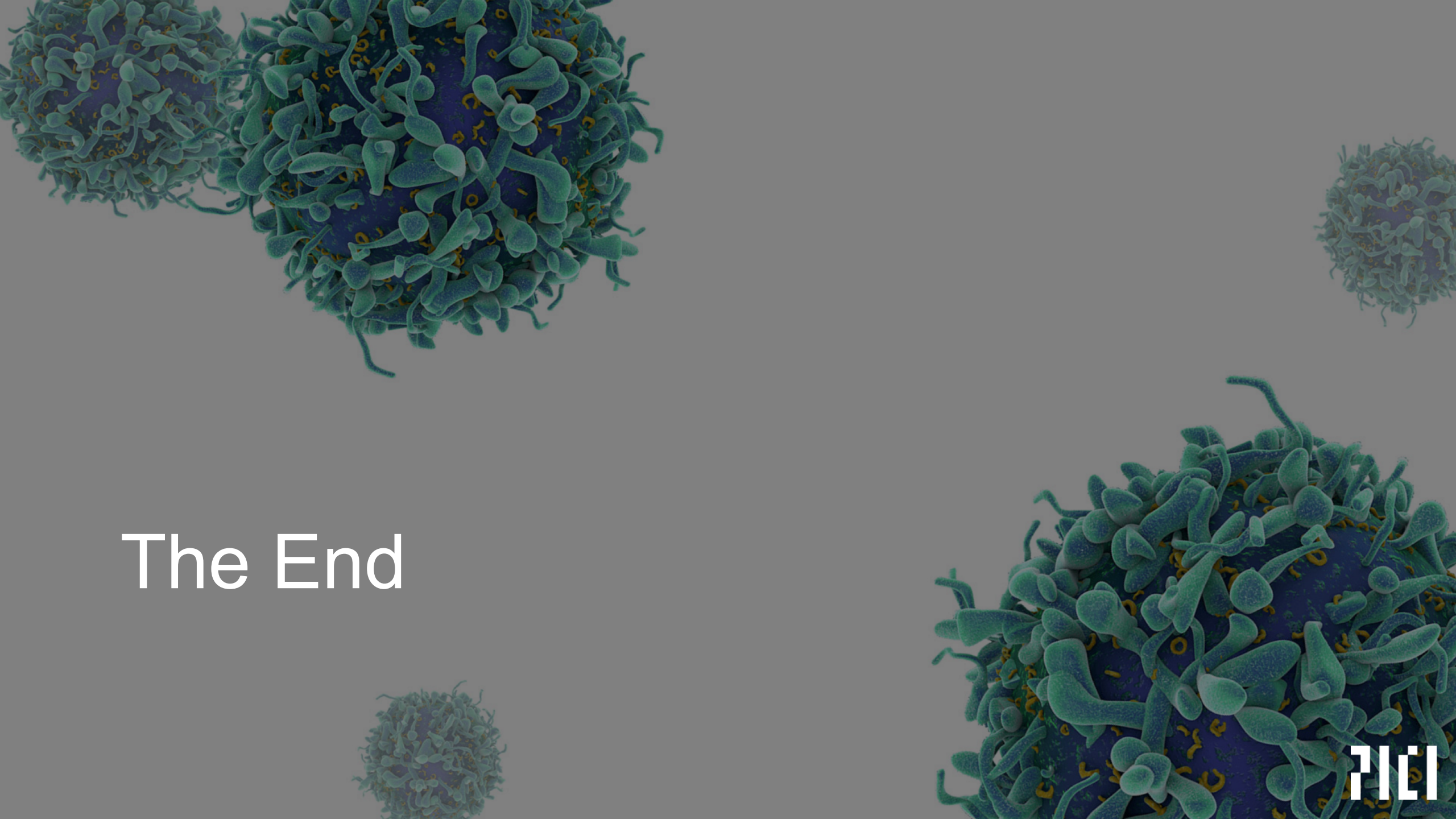
Afferent, 1999



BioBike, 2006



Enflame, 2019


The End