CSAIL

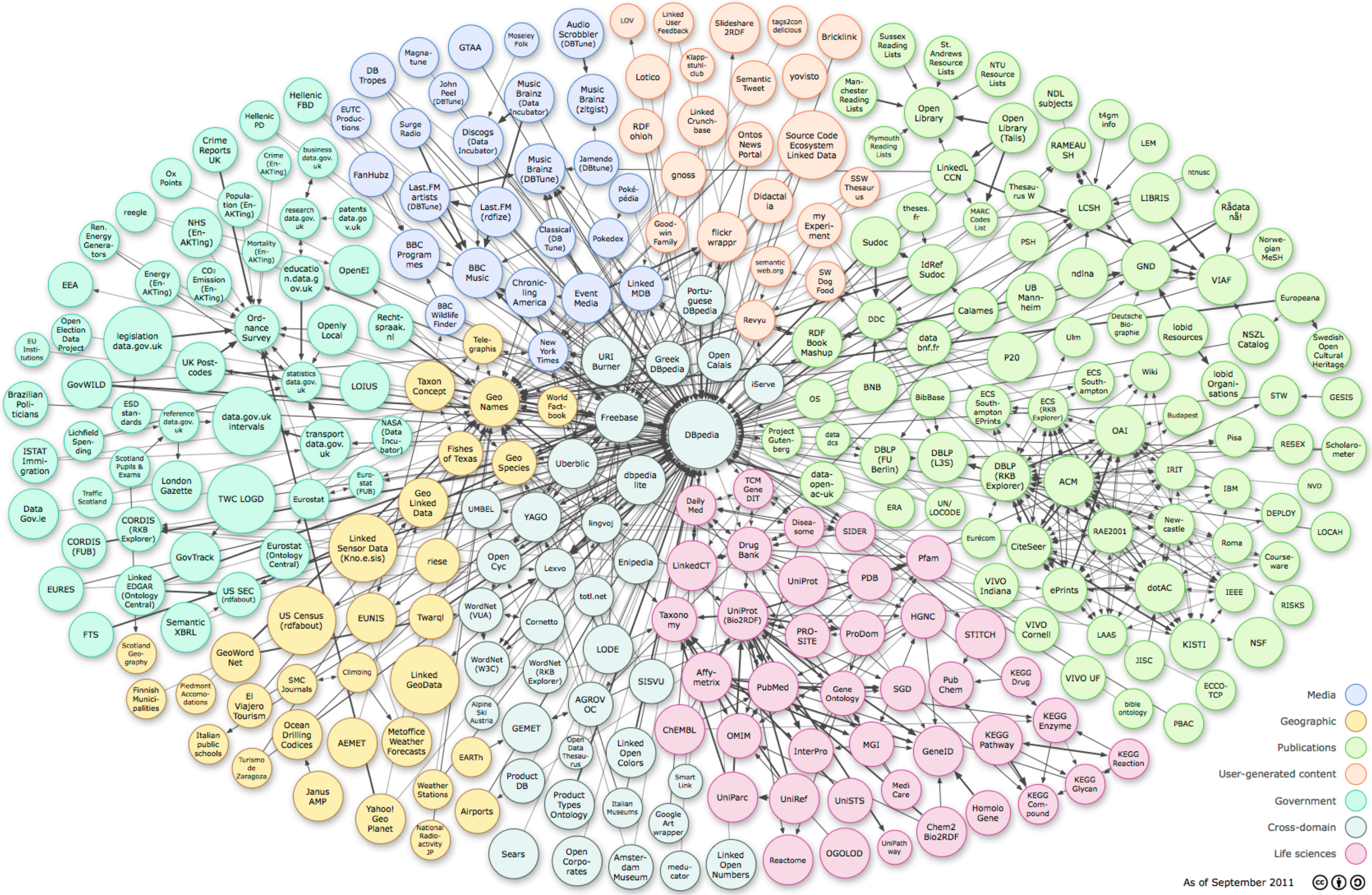
MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY

# Linked Data Connections in App Inventor

Lalana Kagal, Evan Patton, Fuming Shih, and **Oshani Seneviratne**

**(oshani@mit.edu)**

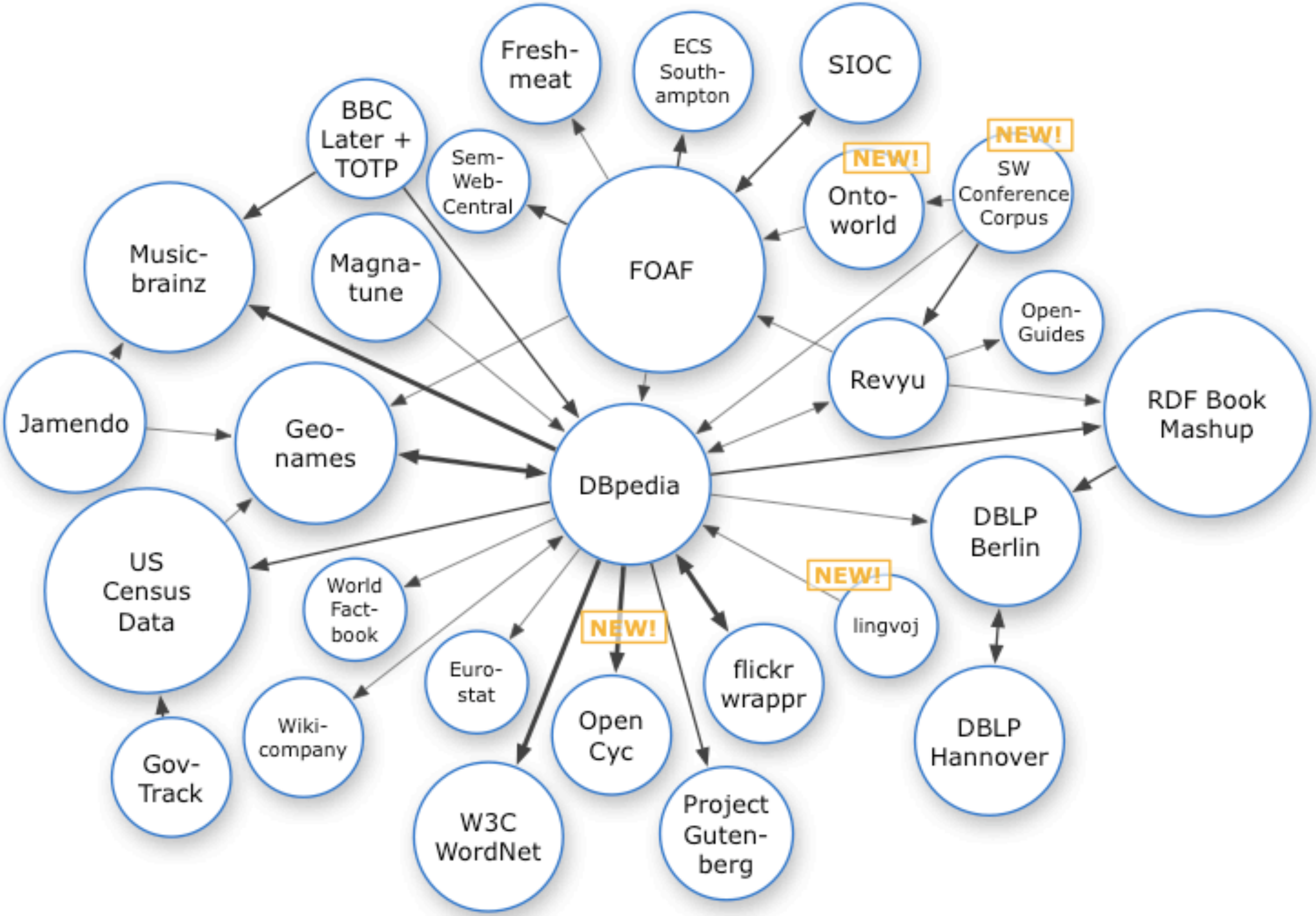
# Linked Data

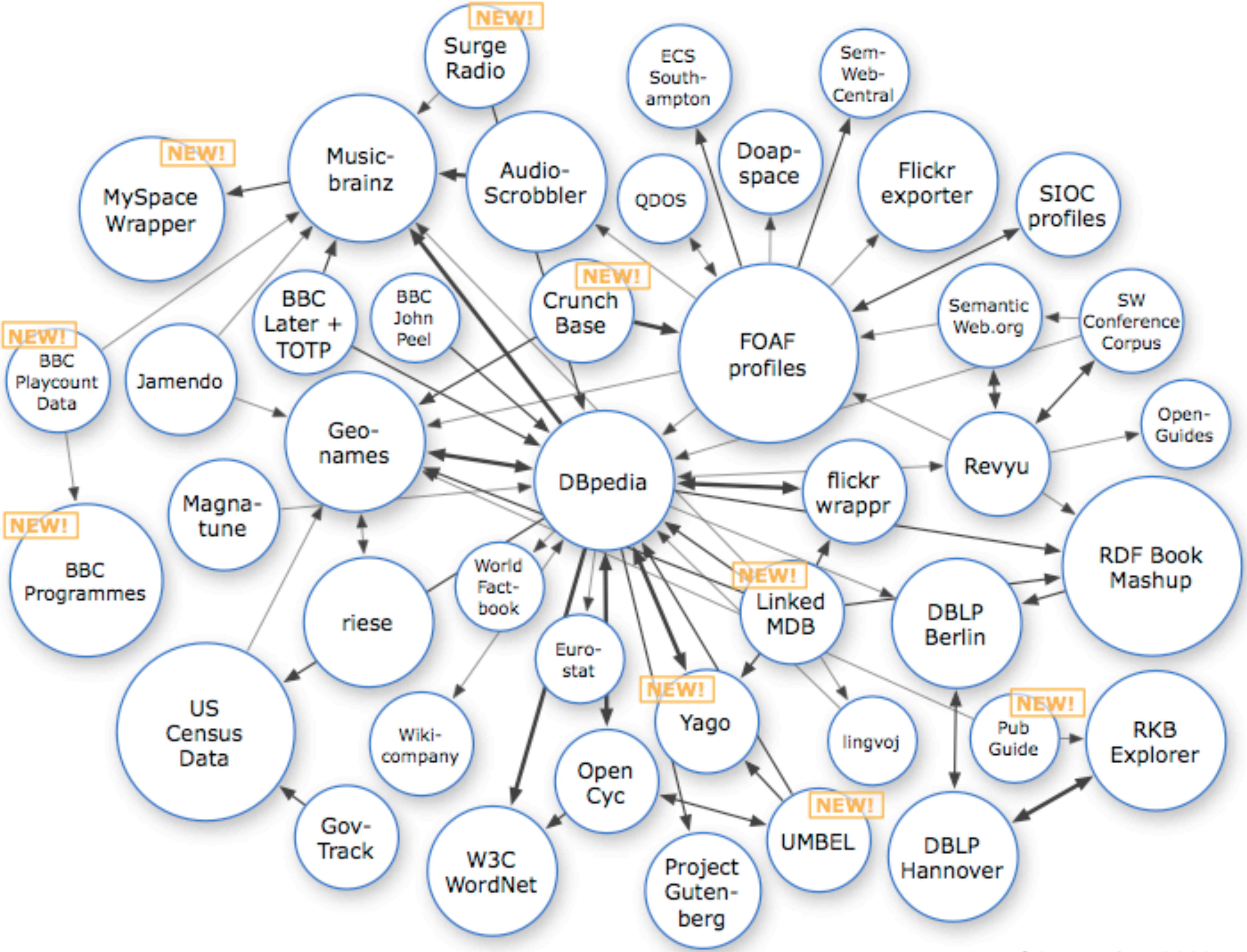


As of September 2011

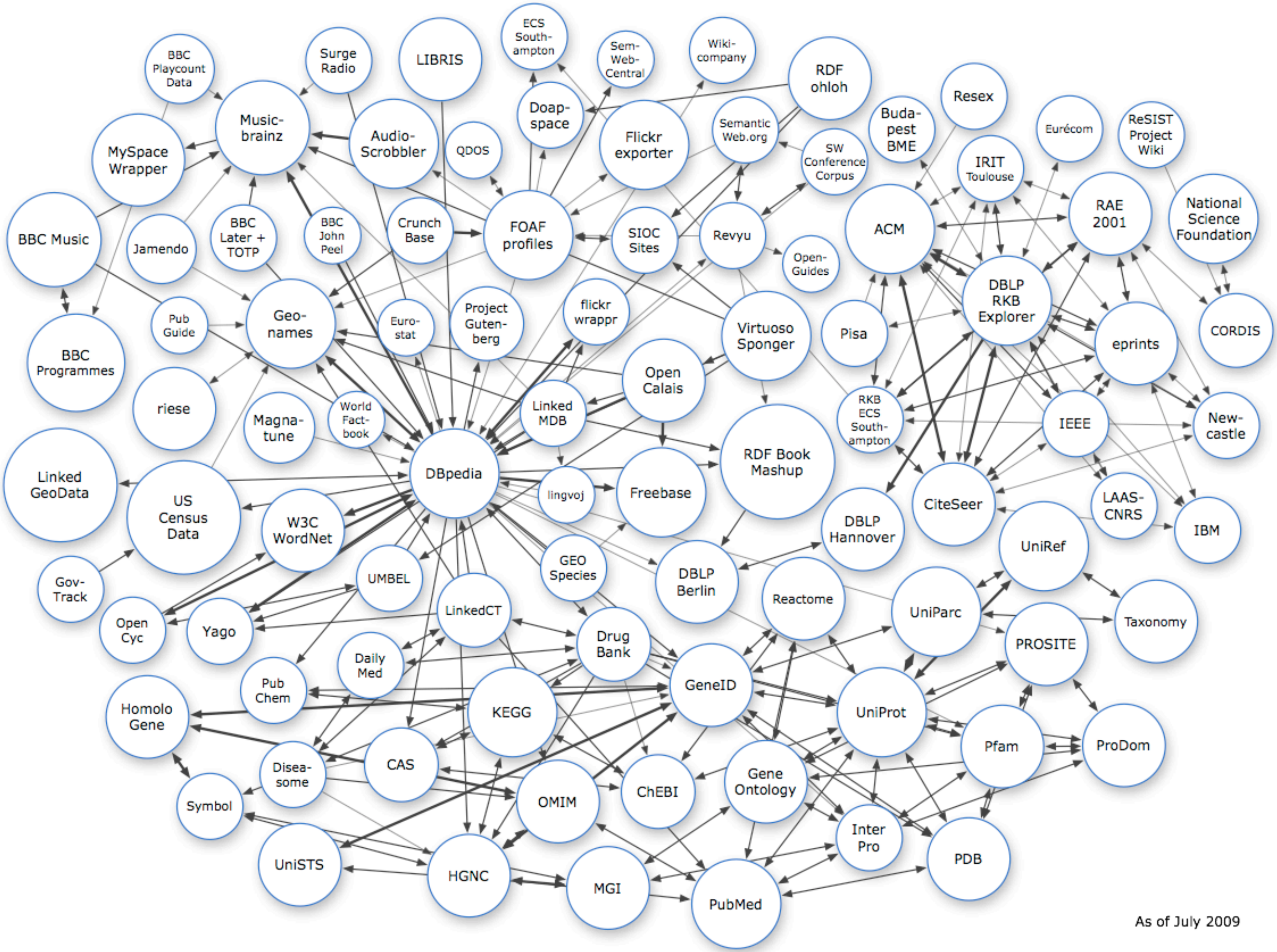
# Linked Data Evolution

2007 → now

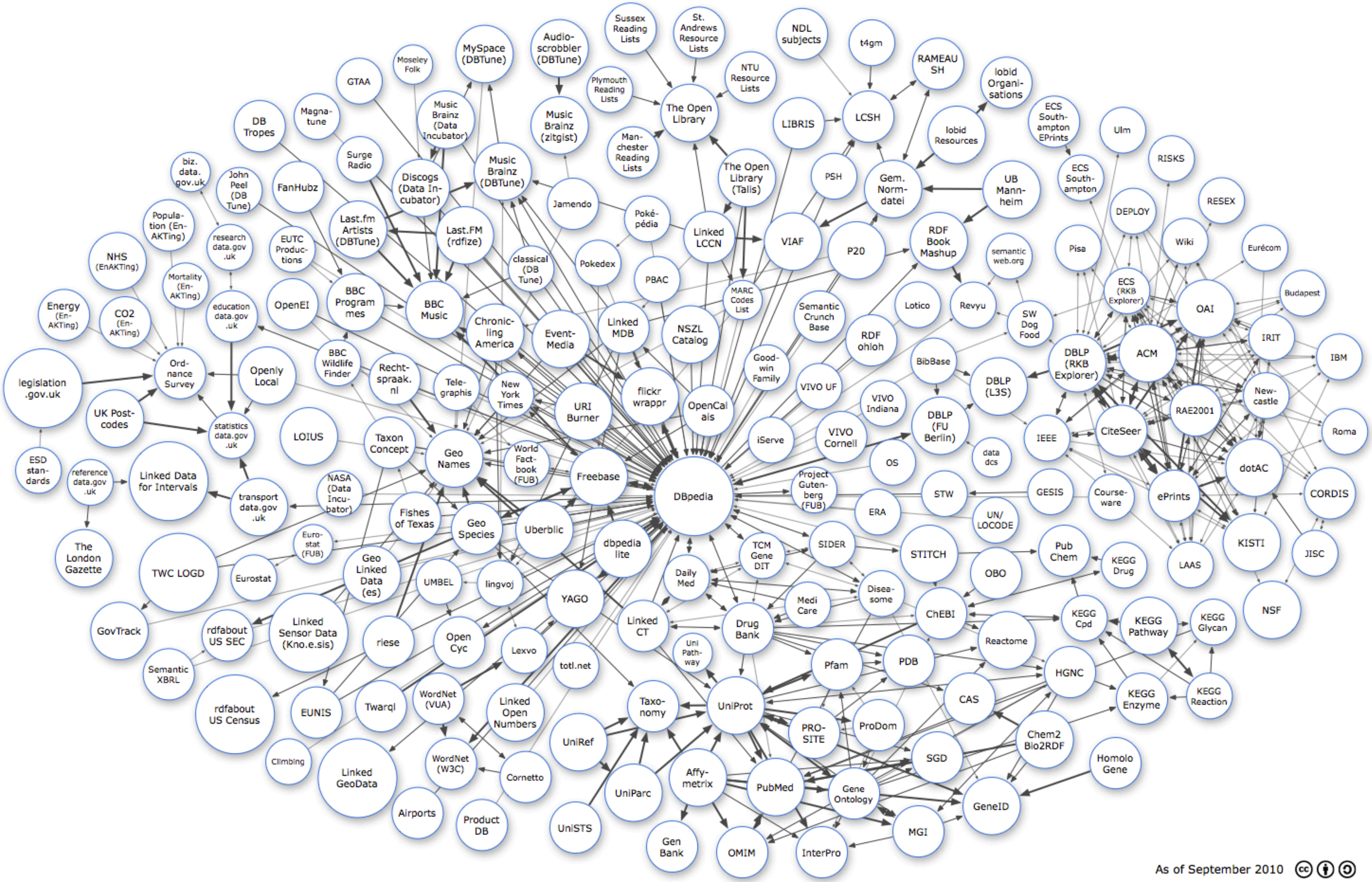

Fresh-meat
ECS South-ampton
SIOC
BBC Later + TOTP
Sem-Web-Central
NEW!
Onto-world
NEW!
SW Conference Corpus
Music-brainz
Magna-tune
FOAF
Open-Guides
Revyu
RDF Book Mashup
Jamendo
Geo-names
DBpedia
DBLP Berlin
US Census Data
World Fact-book
NEW!
lingvoj
Euro-stat
NEW!
Open Cyc
flickr wrappr
DBLP Hannover
Gov-Track
Wiki-company
W3C WordNet
Project Guten-berg



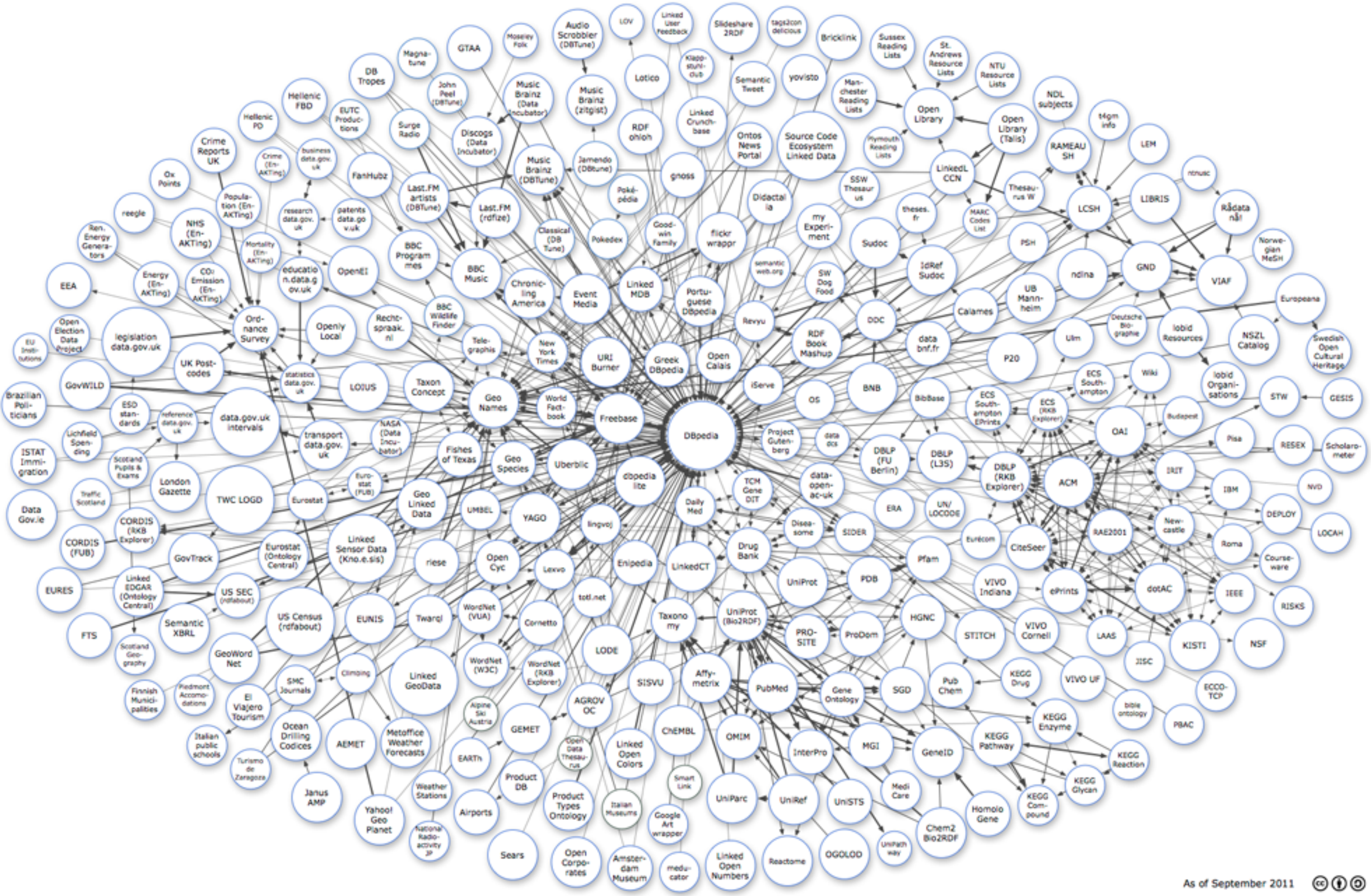
As of September 2008



As of July 2009



As of September 2010



As of September 2011

# Why use Linked Data?

- Enormous wealth of open data

- Uniform Resource Identifiers (URIs) enable interconnections between datasets, unlike data in traditional databases

# How to work with linked data?

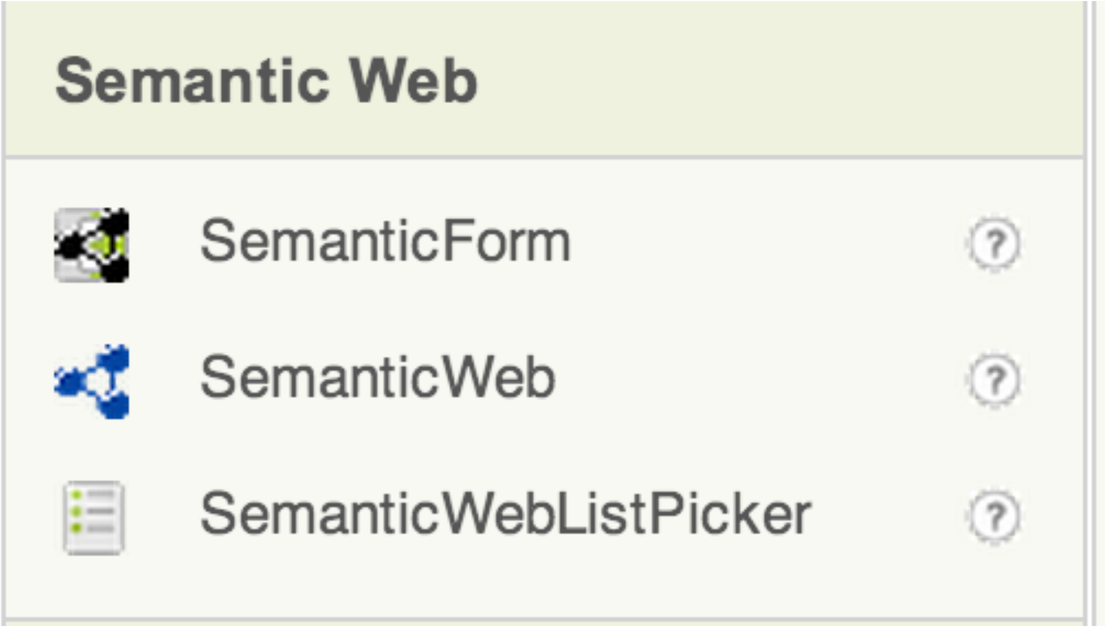
- To consume:
  - Query using SPARQL

- To produce:
  - Create “triples” and/or “graph” structures using Resource Description Framework (RDF)

# Our goals

- Introduce linked data to App Inventor
- Hide the complexity of RDF, SPARQL when developing mobile apps

# The App Inventor Components



**Semantic Web**

SemanticForm

SemanticWeb

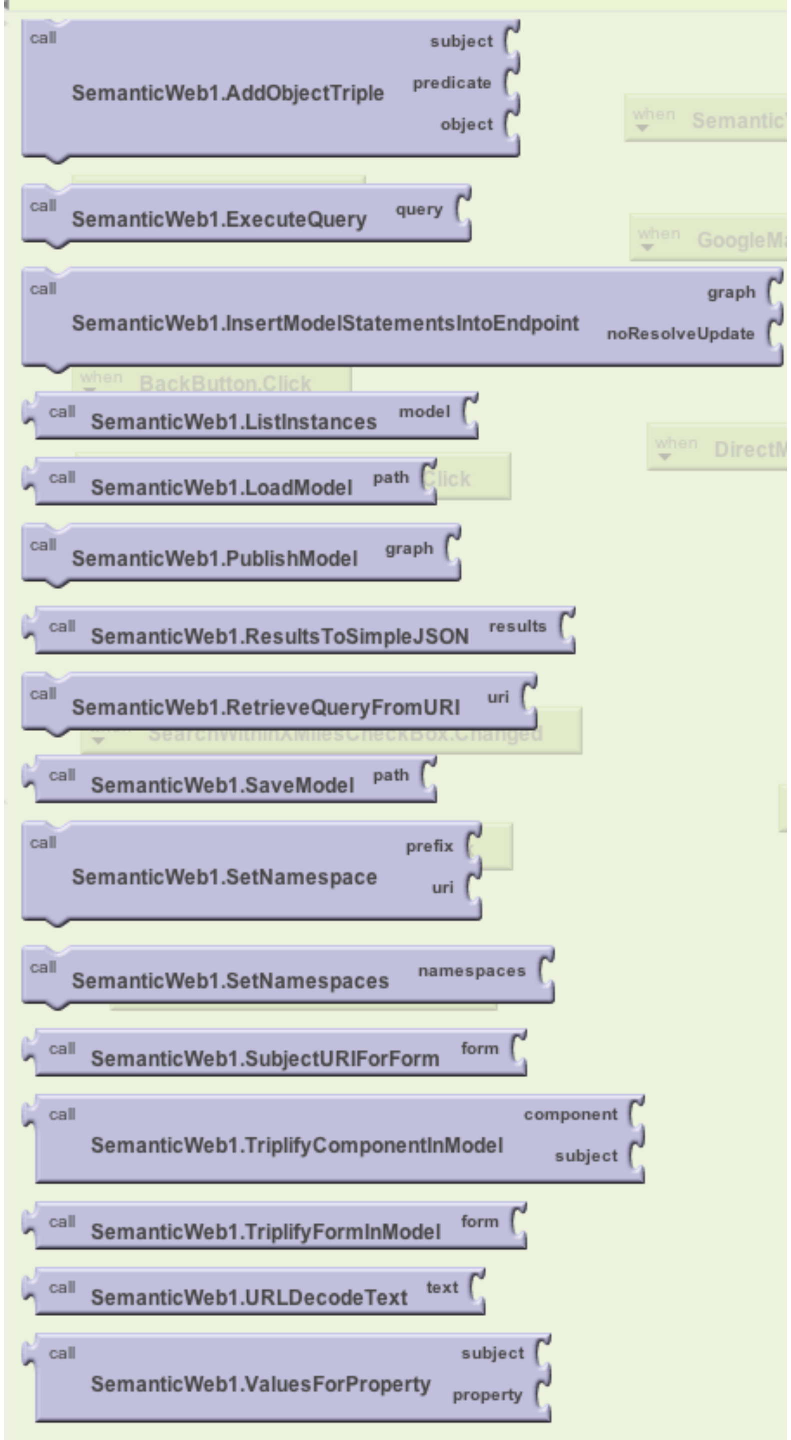
SemanticWebListPicker

# SemanticForm

- A new app inventor layout
- Creates an RDF model based the contents in the form
- Can create more complex structures with nested forms
- The form can mint new URIs using their contents

# SemanticWeb

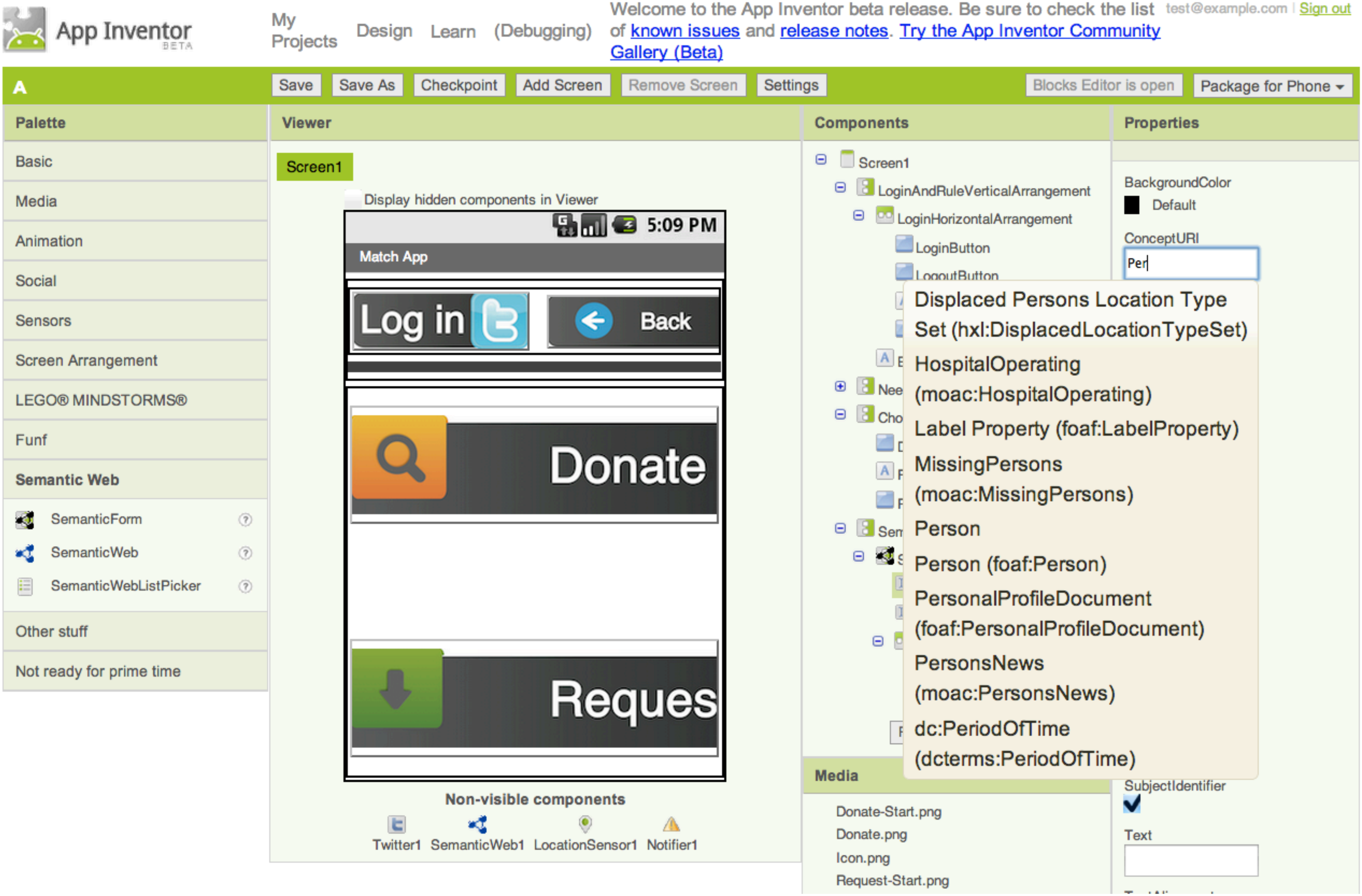
- Provides methods for:
  - Reading RDF
  - Storing and manipulating RDF on the device
  - Writing out an RDF model to a triple store



```
call SemanticWeb1.AddObjectTriple subject predicate object
call SemanticWeb1.ExecuteQuery query
call SemanticWeb1.InsertModelStatementsIntoEndpoint graph noResolveUpdate
call SemanticWeb1.ListInstances model
call SemanticWeb1.LoadModel path
call SemanticWeb1.PublishModel graph
call SemanticWeb1.ResultsToSimpleJSON results
call SemanticWeb1.RetrieveQueryFromURI uri
call SemanticWeb1.SaveModel path
call SemanticWeb1.SetNamespace prefix uri
call SemanticWeb1.SetNamespaces namespaces
call SemanticWeb1.SubjectURIForForm form
call SemanticWeb1.TriplifyComponentInModel component subject
call SemanticWeb1.TriplifyFormInModel form
call SemanticWeb1.URLDecodeText text
call SemanticWeb1.ValuesForProperty subject property
```

# SemanticWebListPicker

- Extends the List Picker component
- Auto-populates contents from a triple store



Welcome to the App Inventor beta release. Be sure to check the list of known issues and release notes. Try the App Inventor Community Gallery (Beta)

BackgroundColor: Default

ConceptURI: Per

SubjectIdentifier: ✓

Text:

# “MatchApp” Demo

- A mobile application that can be easily built using disaster scenarios
- Donors can input information about donations
- People in the disaster areas can search for those resources and request them via Twitter

# Check us out!

Source Code:

https://github.com/mit-dig/appinventor-sources

DIY Mobile App Development for Disaster Management

http://dig.csail.mit.edu/2013/QCRI-DIG-project

# Thank You!