Gist Tutorials

Gist is both a free reusable asset that you can use as the basis for your ontological work. It is also the distillation of a decade worth of ontological work at the enterprise level.

We’ve prepared a series of brief tutorials each of which describes some aspect of gist. Some of the early ones focus more on what gist is, and how to use it, as we get deeper in there will be a lot of commentary on why gist is the way it is.

We’ll be releasing new modules periodically (which loosely means when we get around to it). If you like to be notified when we put a new one up here:

[ email box] [button: Let me know when you post a new tutorial]

Gist 7 tutorials

* Getting Started with Gist 7 – we cover three different ways to download and start working with gist.
* Grokking gist – In this tutorial we cover “getting your head around” gist.
* Gist Modularity – gist consists of 18 very small modules. This tutorial outlines what they are and what their dependencies are
* Gist Primitives – The core of gist are a dozen high level concepts that are mostly disjoint from each other. This tutorial covers what they are and why they have settled in the way they have.
* Concrete Abstractions – one of the key concepts in gist is the idea of basing the ontology on “concrete abstractions” by that we mean things that are relatively abstract, but still give rise to instances that correspond directly to real world things.
* Gist as an Integration Platform – the gist concepts cover most of the concepts you’ll find in most enterprise systems. As a result they provide a great starting point for rationalizing your integration efforts.
* Gist as a Development Platform – this tutorial outlines the thought process you’d go through if you’re using gist as the basis for an application ontology or directly application building
* Gist v. Other Upper Ontologies – in this section we look at some other upper level ontologies that exist that you may consider using in your ontology building
* Gist namespaces and version naming -- we’ve adopted conventions for namespaces and versioning that are helpful to know if you’re using gist, and might be reasonable to emulate when you create your own ontologies
* Integrating gist with other popular ontologies – in this section we describe our approach to aligning with other ontologies such as Dublin Core, Void, Skos and the like. Each will be covered in more detail in a specific tutorial.
* Gist and Time – includes how gist represents time, and we’ll discuss where gist fits into the endurant/ pedurant and 3D/4D time debate. We briefly cover our approach to temporal relationships.
* Gist Places – gist uses a small number of primitives to represent the concepts you’d find in maps and in GPS and GIS systems
* Gist People – we’ll discuss the gist concept of person and contrast it with the FOAF concept of person (spoiler alert: in gist Shrek is not a Person)
* Gist Organizations – most ontologies make commitments about organizations that don’t hold in all the cases we’ve encountered. We discuss the minimalist representation of Organization and how you can specialize it if appropriate in narrower domains
* Gist Units of Measure – we have adopted a simple but complete version of the Society International system of measures, which base all units in seven primitives (weight, distance, time duration, temperature, electric current, luminescence and of course molar quantity (basis for avagadros number for those who remember their Chemistry). We’ve extended it with ratios (speed) products (force) currency and count.
* Gist Magnitude – all measures in gist are what we call magnitudes, and because of their tight coupling with gist Units we are able to support any arbitrary unit conversion.
* Gist Events, Tasks and Behaviors – we explain how we handle everything from project management to work flow to business process and much more in a few simple primiitives
* Gist Addresses – we take the position that addresses are not attributes of people or organizations but are first class objects in their own right. Many great things are made possible y this simple re-conceptualization
* Gist Identifiers – in Semantics all things have uri’s as their primary identifier, but because semantic systems live in the broader context of other systems we’ve created a way to express the myriad of identifiers all systems have to deal ith
* Gist Physical Items and Substances – In a very few classes we resolve the dilemma of the statute and the bronze.
* Gist Content – here we have the world of documents, audio, video images and the like. We also deal with “works” in a fashion similar to FBRL.
* Gist Agreements and Obligations – turns out most of most business deal with the establishment, management and discharging of obligations.
* Gist Categories, Controlled Vocabularies and Taxonomies – One of the secrets to having an ontology that is stable and dynamic is knowing which parts are foundational and which parts are more arbitrary categorizations that can be delegated to governance groups and industry standards.
* Gist Intention – this is the basis for the “future” for most concepts. Intention, goals, requirement and restrictions are all things that operate on objects and commitments in the future. This is the basis for the teleological part of gist.
* Gist Maps – this is a place holder for a whole series of specific maps to other ontologies.