Reasoning in JSON-LD?

- Leverage use of anonymously named graphs to do implication (**log:implies**).
  - Requires a way to identify universal quantifiers (existential quantifiers simply blank nodes)

Notation3

```
@forAll :x, :y.
:Julie :parent :Suzie .
{ :x :parent :y } =>  { :y :child :x }.
```

```
:Julie :parent :Suzie .
{ ?x parent ?y } =>  { ?y :child ?x }.
```

JSON-LD

```
{
  "@context": {
    "@base": "http://example.com/",
    "@vocab": "http://example.com/",
    "=>": {"@id": "http://www.w3.org/2000/10/swap/log#implies", "@container": "@graph"},
    "?x": {"@type": "@univar"},
    "?y": {"@type": "@univar"}
  },
  "@graph": [
    {"@id": "Julie", "parent": {"@id": "Suzie"}},
    {
      "@graph": {"@id": "?x", "parent": "?y"},
      "=>": {"@id": "?y", "child": "?x"}
    }
  ]
}
```

Note @univar is totally hypothetical

Note hand waiving on how “?y” is expanded

JSON-LD 1.1

- It's been over three years since JSON-LD 1.0 was published, and feature requests have been mounting:
  - 36 issues addressed since 1.0 (15 still open)
  - Use objects to index into collections, rather than only array form
    - Previously restricted to `@index` and `@language`. Now available on `@id` and `@type`.
    - Can include `@set` with other container types (e.g.: `"@container": ["@set", "@language"]`).
  - Framing, never complete in 1.0. Now provides ability to match on `@id`, inclusive or exclusive `@type`, property values, and specifics of a value object. Supports framing of datasets, not just graphs.
  - Contexts scoped to terms: property values or entities using a given type term can overlay terms-specific contexts.
  - Ignore some elements of JSON structure.
  - Abstract from JSON-itself, allowing for YAML, CBOR and other LD representations.