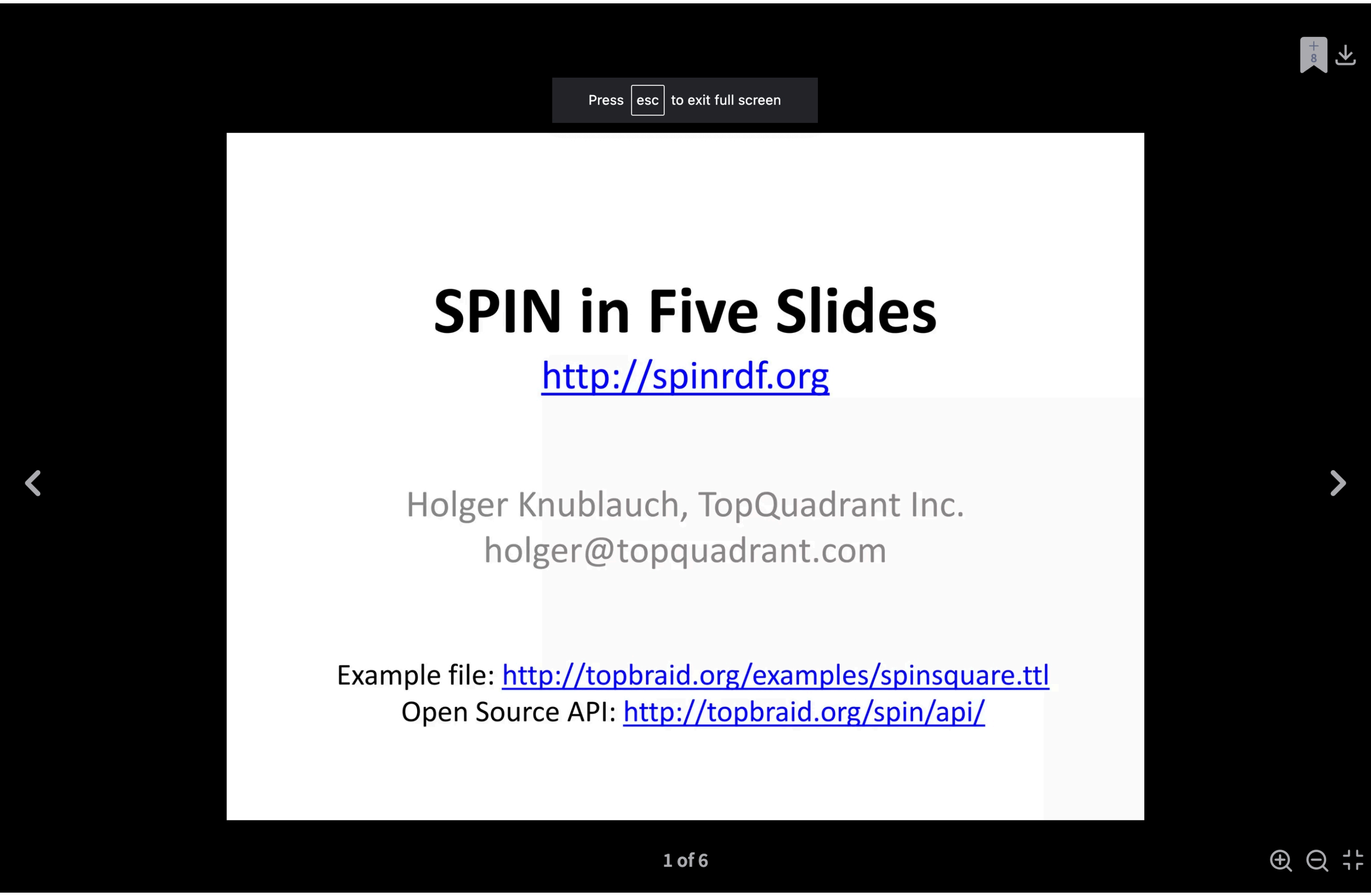
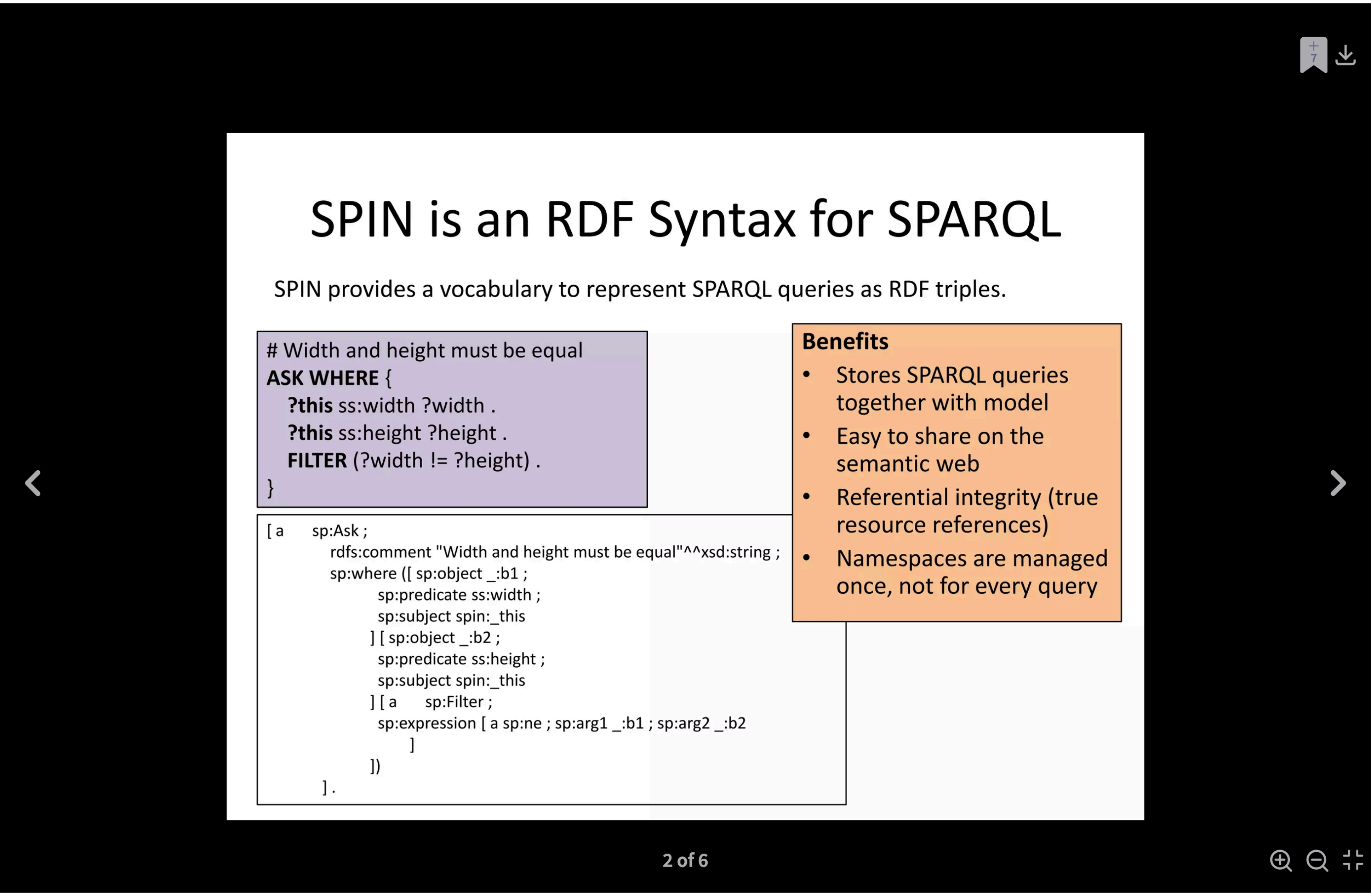


SPIN in Five Slides

http://spinrdf.org

Holger Knublauch, TopQuadrant Inc.
holger@topquadrant.com

Example file: http://topbraid.org/examples/spinsquare.ttl
Open Source API: http://topbraid.org/spin/api/



SPIN is an RDF Syntax for SPARQL

SPIN provides a vocabulary to represent SPARQL queries as RDF triples.

```
# Width and height must be equal
ASK WHERE {
   ?this ss:width ?width .
   ?this ss:height ?height .
   FILTER (?width != ?height) .
}
```

```
[ a      sp:Ask ;
         rdfs:comment "Width and height must be equal"^^xsd:string ;
         sp:where ([ sp:object _:b1 ;
                sp:predicate ss:width ;
                sp:subject spin:_this
              ] [ sp:object _:b2 ;
                sp:predicate ss:height ;
                sp:subject spin:_this
              ] [ a      sp:Filter ;
                sp:expression [ a sp:ne ; sp:arg1 _:b1 ; sp:arg2 _:b2
                     ]
              ])
        ] .
```

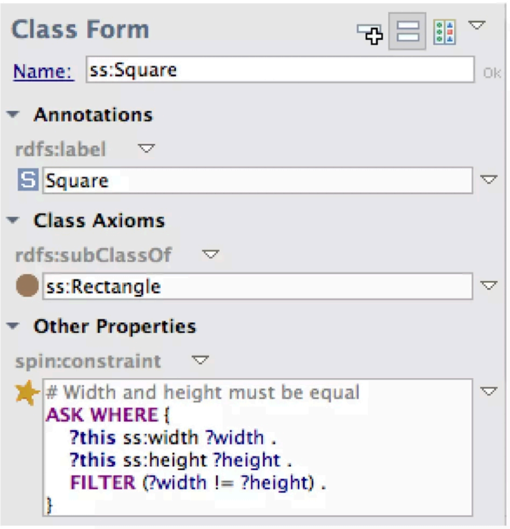
Benefits

- Stores SPARQL queries together with model
- Easy to share on the semantic web
- Referential integrity (true resource references)
- Namespaces are managed once, not for every query

2 of 6

SPIN is a SPARQL Constraint Language

The property spin:constraint can be used to link a class with SPARQL queries that formalize invariants for the members of that class.



Class Form

Name: ss:Square

Annotations

rdfs:label

Square

Class Axioms

rdfs:subClassOf

ss:Rectangle

Other Properties

spin:constraint

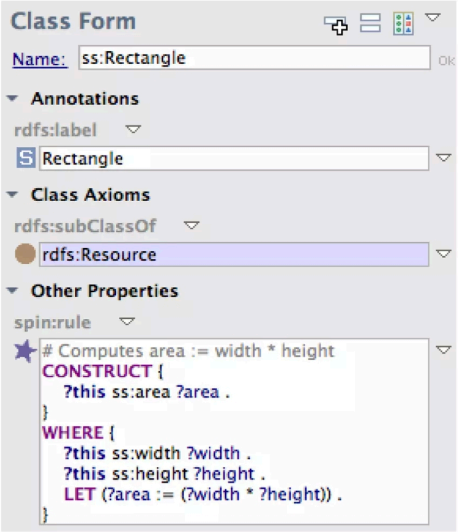
```
# Width and height must be equal
ASK WHERE {
   ?this ss:width ?width .
   ?this ss:height ?height .
   FILTER (?width != ?height) .
}
```

Benefits

- Natural object-oriented way of modeling
- SPARQL is very expressive
- Constraints can be natively executed by SPARQL engines of the database
- Easy to combine with other SPARQL constraint bases like SKOS SPIN

SPIN is a SPARQL Rules Language

The property spin:rule can be used to link a class with SPARQL CONSTRUCT queries that define inference rules for the members of the class



Class Form

Name: ss:Rectangle

Annotations

rdfs:label

S Rectangle

Class Axioms

rdfs:subClassOf

rdfs:Resource

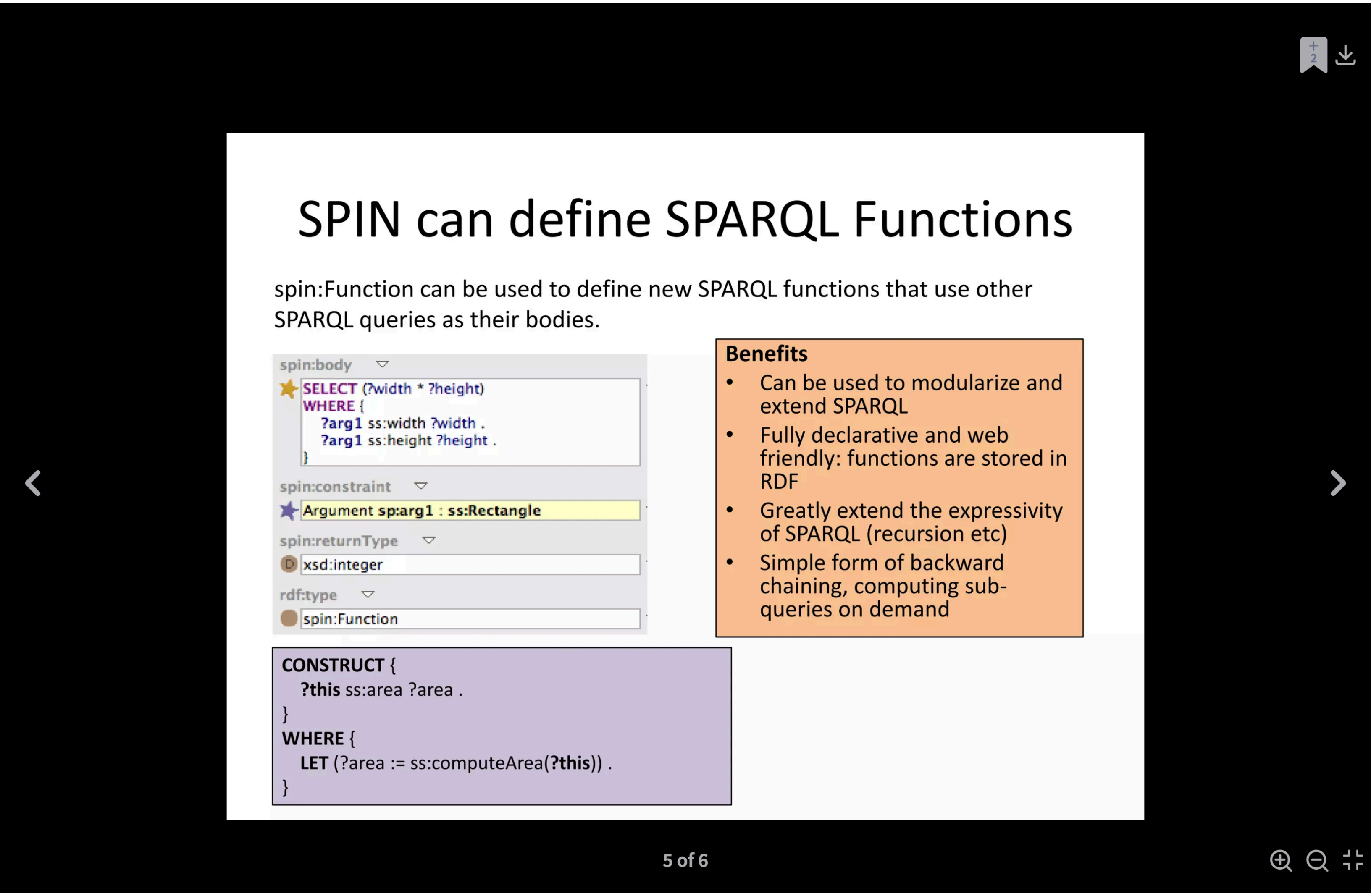
Other Properties

spin:rule

```
# Computes area := width * height
CONSTRUCT {
   ?this ss:area ?area .
}
WHERE {
   ?this ss:width ?width .
   ?this ss:height ?height .
   LET (?area := (?width * ?height)) .
}
```

Benefits

- Natural object-oriented way of modeling
- SPARQL is very expressive
- Rules can be natively executed by SPARQL engines of the database
- Easy to combine with other SPARQL rule bases like OWL RL



SPIN can define SPARQL Functions

spin:Function can be used to define new SPARQL functions that use other SPARQL queries as their bodies.

spin:body

```
SELECT (?width * ?height)
WHERE {
    ?arg1 ss:width ?width .
    ?arg1 ss:height ?height .
}
```

spin:constraint

Argument sp:arg1 : ss:Rectangle

spin:returnType

xsd:integer

rdf:type

spin:Function

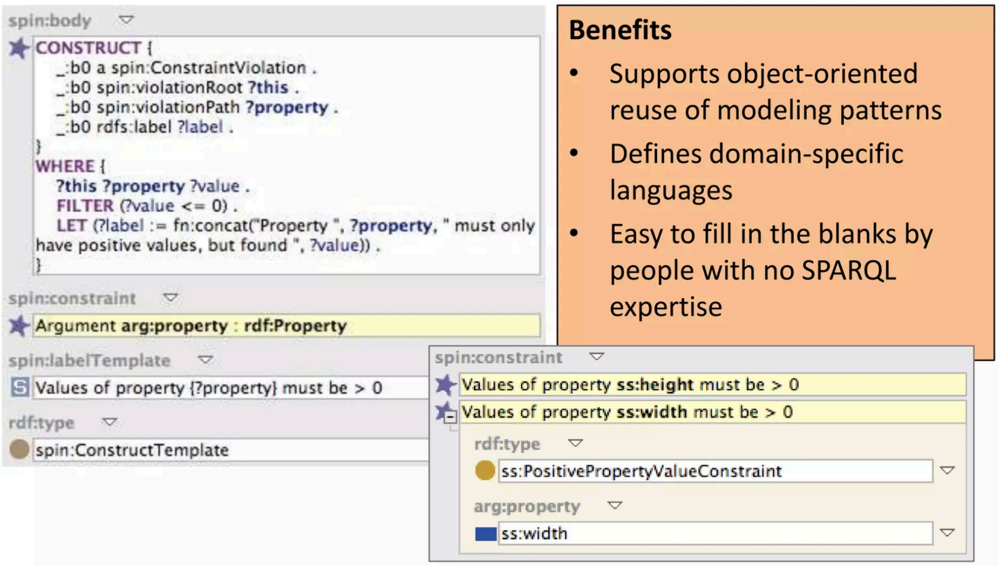
Benefits

- Can be used to modularize and extend SPARQL
- Fully declarative and web friendly: functions are stored in RDF
- Greatly extend the expressivity of SPARQL (recursion etc)
- Simple form of backward chaining, computing sub-queries on demand

```
CONSTRUCT {
    ?this ss:area ?area .
}
WHERE {
    LET (?area := ss:computeArea(?this)) .
}
```

SPIN can store reusable SPARQL queries

spin:Template can be used to wrap SPARQL queries into reusable building blocks so that they can be instantiated with arguments



spin:body

```
CONSTRUCT {
    _:b0 a spin:ConstraintViolation .
    _:b0 spin:violationRoot ?this .
    _:b0 spin:violationPath ?property .
    _:b0 rdfs:label ?label .
}
WHERE {
    ?this ?property ?value .
    FILTER (?value <= 0) .
    LET (?label := fn:concat("Property ", ?property, " must only
have positive values, but found ", ?value)) .
}
```

spin:constraint

Argument **arg:property** : **rdf:Property**

spin:labelTemplate

Values of property {?property} must be > 0

rdf:type

spin:ConstructTemplate

spin:constraint

Values of property **ss:height** must be > 0

Values of property **ss:width** must be > 0

rdf:type

ss:PositivePropertyValueConstraint

arg:property

ss:width

Benefits

- Supports object-oriented reuse of modeling patterns
- Defines domain-specific languages
- Easy to fill in the blanks by people with no SPARQL expertise