ShEx:
Shape Expressions language

Kat Thornton, Tom Baker, Andra Waagmeester, Eric Prud'hommeaux

SWIB-2018

Bonn, 2018-11-26

XML Schema

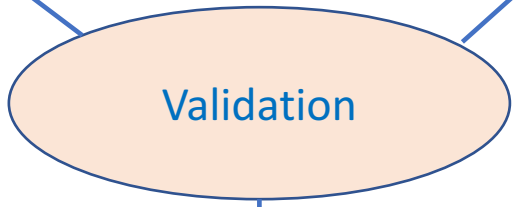
```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

   <xsd:annotation>
      <xsd:documentation xml:lang="en">
       Purchase order schema for Example.com.
       Copyright 2000 Example.com. All rights reserved.
      </xsd:documentation>
   </xsd:annotation>

<xsd:element name="purchaseOrder" type="PurchaseOrderType"/>

<xsd:element name="comment" type="xsd:string"/>

<xsd:complexType name="PurchaseOrderType">
   <xsd:sequence>
      <xsd:element name="shipTo" type="USAddress"/>
```

XML Data

```
<?xml version="1.0"?>
<purchaseOrder orderDate="1999-10-20">
   <shipTo country="US">
      <name>Alice Smith</name>
      <street>123 Maple Street</street>
      <city>Mill Valley</city>
      <state>CA</state>
      <zip>90952</zip>
   </shipTo>
   <billTo country="US">
      <name>Robert Smith</name>
      <street>8 Oak Avenue</street>
      <city>Old Town</city>
      <state>PA</state>
      <zip>95819</zip>
   </billTo>
```

2. Against a schema.

1. Validate data.



3. To yield a validation result

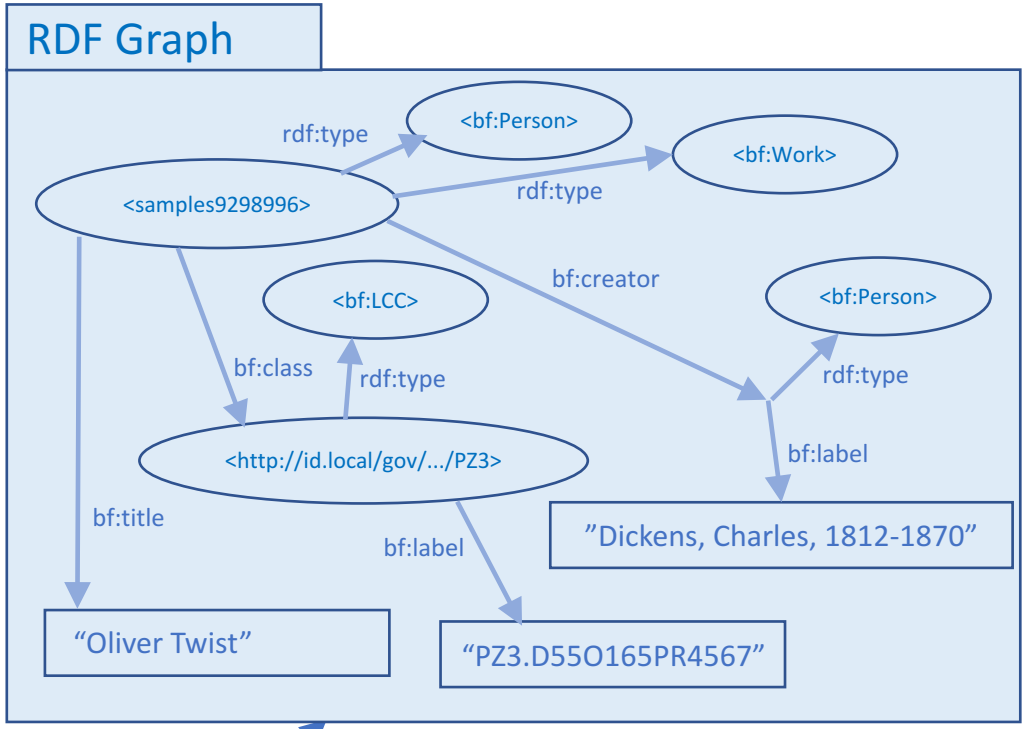
D:\data\purchase_order23.xml is valid.

XML Schema Definition language: schema prescribes conditions to which an XML document must conform to be considered valid.

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Graph



2. Against a schema.

1. Validate graph.

Validation

Shape Expressions language (ShEx): schema prescribes conditions to which an RDF graph must conform to be considered valid.

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

Shape Expressions language (ShEx): schema prescribes conditions to which RDF data must conform to be considered valid.

ShEx Schema [ShExJ syntax]

```
{
  "type": "Schema",
  "shapes": [
    {
      "id": "https://rawgit.com/shexSpec/shex.js/master/doc/Work",
      "type": "Shape",
      "expression": {
        "type": "EachOf",
        "expressions": [
          {
            "type": "TripleConstraint",
            "predicate": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type",
            "valueExpr": {
              "type": "NodeConstraint",
              "values": [
                "http://bibframe.org/vocab/Work"  ... ... ...
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/.../PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/.../PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

Validation

ShExJ syntax, interchangeable with ShExC.

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource in RDF data matching the "Work" shape[1]

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

[1] Note: a pre-processing step uses the ShEx ShapeMap language to associate an RDF nodes with ShEx shapes, here: **`<samples9298996>@<Work>`**. Associations can be enumerated (in "fixed shape maps") or generated automatically (in "query shape maps") depending on the degree of control desired.

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource in RDF data matching the “Work” shape
Has zero or one “rdf:type bf:Work” statements

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource in RDF data matching the “Work” shape:
Has zero or one “rdf:type bf:Work” statements
Optionally has an extra “rdf:type” statement

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource in RDF data matching the “Work” shape:
Has zero or one “rdf:type bf:Work” statements
Optionally has an extra “rdf:type” statement
Has exactly one “bf:title” statement with literal value

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource in RDF data matching the “Work” shape:
Has zero or one “rdf:type bf:Work” statements
Optionally has an extra “rdf:type” statement
Has exactly one “bf:title” statement with literal value
Has zero or more “bf:class” statements taking objects that match the “Classification” shape

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

Validation

A resource in RDF data matching the “Work” shape:
Has zero or one “rdf:type bf:Work” statements
Optionally has an extra “rdf:type” statement
Has exactly one “bf:title” statement with literal value
Has zero or more “bf:class” statements taking objects that match the “Classification” shape
Has one or more “bf:creator” statements taking objects that match either the “Person” or the “Organization” shape

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

Validation

3. To yield a validation result

A resource in RDF data matching the “Work” shape:
Has zero or one “rdf:type bf:Work” statements
Optionally has an extra “rdf:type” statement
Has exactly one “bf:title” statement with literal value
Has zero or more “bf:class” statements taking objects that match the “Classification” shape
Has one or more “bf:creator” statements taking objects that match either the “Person” or the “Organization” Shape
And zero or more “bf:derivedFrom” statements taking an IRI as object.

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource matching the "Classification" shape:

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

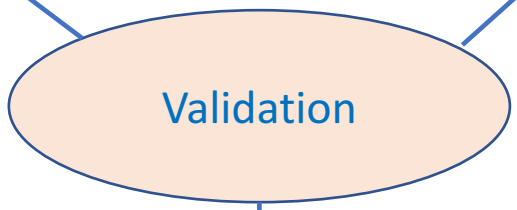
RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<https://id.loc.gov/authorities/classification/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource matching the “Classification” shape:
is identified with an IRI starting with “id.loc.gov”



3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
  validating http://...oliverTwist:
    Error validating http://...oliverTwist
    as nodeKind literal:
      iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource matching the “Classification” shape:
is identified with an IRI starting with “id.loc.gov”
has zero or one “rdf:type bf:LOC” statements

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource matching the “Classification” shape:
is identified with an IRI starting with “id.loc.gov”
has zero or one “rdf:type bf:LOC statements
optionally has an additional “rdf:type” statement

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

A resource matching the “Classification” shape:
is identified with an IRI starting with “id.loc.gov”
has zero or one “rdf:type bf:LOC statements
optionally has an additional “rdf:type” statement
has exactly one “bf:label” statement with a literal value

Validation

3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
   validating http://...oliverTwist:
      Error validating http://...oliverTwist
      as nodeKind literal:
        iri found when literal expected
```

ShEx Schema [ShExC syntax]

```
<Work> EXTRA rdf:type {
  rdf:type [bf:Work] ? ;
  bf:title LITERAL ;
  bf:class @<Classification> * ;
  bf:creator @<Person> OR @<Organization> + ;
  bf:derivedFrom IRI * ;
}

<Classification>
  [<http://id.loc.gov/…/>~]
AND
  EXTRA rdf:type {
    rdf:type [bf:LCC] ? ;
    bf:label LITERAL ;
}
```

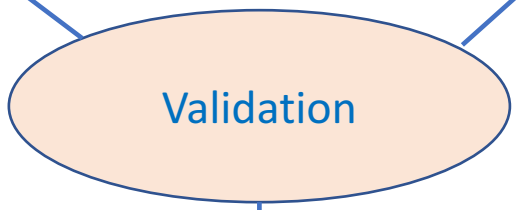
RDF Data [Turtle syntax]

```
<samples9298996>
  rdf:type bf:Text ;
  rdf:type bf:Work ;
  bf:title "Oliver Twist." ;
  bf:class <id.loc.gov/…/PZ3> ;
  bf:creator [
    rdf:type bf:Person ;
    bf:label "Dickens, Charles, 1812-1870." ;
  ] .

<id.loc.gov/…/PZ3>
  rdf:type bf:LCC ;
  bf:label "PZ3.D55O165PR4567" .
```

2. Against a schema.

1. Validate data.

Shapes can combine constraints with AND, OR, NOT, and parentheses.



3. To yield a validation result

✓<samples9298996>@<Work>

✗<samples9298996bad>@!<Work>

```
validating samples9298996bad as Work:
  validating http://...oliverTwist:
    Error validating http://...oliverTwist
    as nodeKind literal:
      iri found when literal expected
```

Terminology summarized

- A **ShEx Schema** holds a collection of...
  - **Shape Expressions**, which logically combine...
    - **Node Constraints**, which match nodes in RDF Data, and...
    - **Shapes**, which match triples in RDF Data as specified in...
      - **Triple constraints**, which match triples touching a given node in RDF Data in terms of...
        - Predicates
        - Direction
        - Cardinality
        - Value

Active and friendly development community

- W3C Shape Expressions Community Group (ShEx CG)
  - https://www.w3.org/community/shex CG homepage
  - http://shex.io ShEx homepage
  - http://shex.io/shex-primer/ primer
  - http://shex.io/shex-semantics/ main specification
  - https://rawgit.com/shexSpec/shex.js/master/doc/shex-simple.html validator

Focus Node and Neighborhood of a Node

- **Focus Node** = node that is being validated (here :alice)
- **Neighborhood of a node** = set of incoming/outgoing triples

```
:alice      schema:name      "Alice";
            schema:follows   :bob;
            schema:worksFor  :OurCompany .

:bob        foaf:name        "Robert" ;
            schema:worksFor  :OurCompany .

:carol      schema:name      "Carol" ;
            schema:follows   :alice .

:dave       schema:name      "Dave" .

:OurCompany schema:founder   :dave ;
            schema:employee  :alice, :bob .
```

```
Neighbourhood of :alice = {
  (:alice,      schema:name,      "Alice")
  (:alice,      schema:follows,   :bob),
  (:alice,      schema:worksFor,  :OurCompany),
  (:carol,      schema:follows,   :alice),
  (:OurCompany, schema:employee,  :alice)
}
```