JSON CONTENT RULES AND RDAP

Andy Newton, Chief Engineer, ARIN

REGOPS Workshop

Buenos Aires, AR

April 2016

RDAP JSON

- Defined in RFC 7483
  - All JSON is specified with prose
  - And lots of examples
- Hard to read
- Imprecise
- Repetitive text
  - Not friendly to native English readers
  - *Really not friendly to non-native English readers*
- No programmatic validators

JSON DDL OR SCHEMA LANGUAGE

- At the time of writing for RFC 7483, non-standardized
  - In the works:
    - JSON Schema
    - JSON Content Rules (JCR)

- Why a JSON DDL?
  - “The Benefits of a JSON Data Definition Language”
    - IETF Journal, April 2016
  - http://www.internetsociety.org/publications/ietf-journal-april-2016/benefits-json-data-definition-language

RDAP JSON IN JCR

- draft-newton-rdap-jcr
  - https://datatracker.ietf.org/doc/draft-newton-rdap-jcr/
- 17 pages
- RFC 7483 – 78 pages

JSON CONTENT RULES

- Looks like JSON

```
{
    "file-name"  : "rfc7159.txt",
    "line-count" : 3426,
    "word-count" : 27886
}
```

LESS SPECIFIC VALUES

- Looks like JSON

```
{
    "file-name"   : string ,
    "line-count" : 0.. ,
    "word-count" : 0..
}
```

RULES

- Rules can be named

```
{
    fn,
    lc,
    wc
}

fn "file-name"   : string
lc "line-count"  : 0..
wc "word-count"  : 0..
```

OVERRIDING RULES

```
{
    fn,
    lc,
    wc
}

fn "file-name"   : string
lc "line-count"  : 0..
wc "word-count"  : 0..
```

```
fn "file-name"   : "rfc4627.txt"
lc "line-count"  : 2102
wc "word-count"  : 16714
```

5 TYPES OF RULES

- Value Rules
  - `v1 : 0..3`
- Member Rules
  - `m1 "m1name" v1`
- Object Rules
  - `o1 { m1, m2 }`
- Array Rules
  - `a1 [ v1, o1 ]`
- Group Rules
  - `g1 ( v1, o1 )`

OTHER FEATURES

- Rules can embed other rules
  - `v1 : 0..3 ; a comment ; m1 "m1name" v1`
  - `m1 "m1name" : 0..3`
- Repetition
  - `a1 [ 1*3 : string, 1*3 : integer ]`
- Choice
  - `o1 { "foo" : string | "bar" : integer }`
- Unordered Arrays
  - `a1 @{unordered} [ : string, : integer ]`
- And more…

CO-CONSTRAINTS

- Takes things even further
- draft-cordell-jcr-co-constraints

```
"type" @{id t} : string

details ( @{when $t == "boot"} boot-details |
          @{when $t == "shutdown"} shutdown-details |
          default-details )
```

RDAP JCR

- Uses the “mix-in” convention

```
nameservers "nameservers" [ * nameserver_oc ]

nameserver_oc {
    nameserver_mixin
}

nameserver_mixin (
      "objectClassName" : "nameserver",
      common_mixin,
      "ldhName"         : fqdn,
    ? "unicodeName"     : idn,
    ? "ipAddresses"     { ? "v4" [ * ip4 ],
                          ? "ip6" [ * ip6 ] },
    ? entities
)
```

MIX-IN CONTINUED

- Nameserver response

```
response_mixin (
    ? rdapConformance,
    ? "notices" notices,
      lang
)

nameserver_response @{root} { response_mixin,
                              nameserer_mixin
}
```

COMMENTS? QUESTIONS?