Metadata schemas express **expectations in the structure of metadata records.**

Customer Information
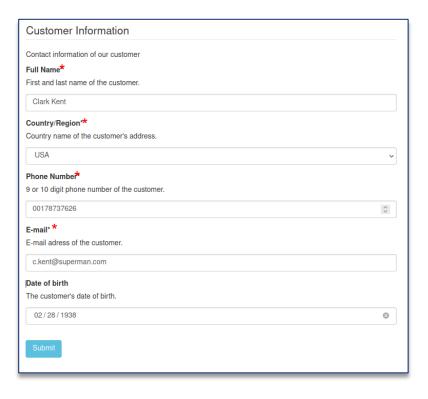
Customer Information

Contact information of our customer

Full Name*
First and last name of the customer.

> Clark Kent

Country/Region*
Country name of the customer's address.

> USA

Phone Number*
9 or 10 digit phone number of the customer.

> 00178737626

E-mail*
E-mail adress of the customer.

> c.kent@superman.com

Date of birth
The customer's date of birth.

> 02 / 28 / 1938

Submit

Customer Information

asterisks indicate mandatory entries

controlled list

entry format

Customer Information

Customer Information

Contact information of our customer

Full Name *
First and last name of the customer.

```
Clark Kent
```

Country/Region *
Country name of the customer's address.

```
USA ▾
```

Phone Number *
9 or 10 digit phone number of the customer.

```
00178737626
```

E-mail *
E-mail adress of the customer.

```
c.kent@superman.com
```

Date of birth
The customer's date of birth.
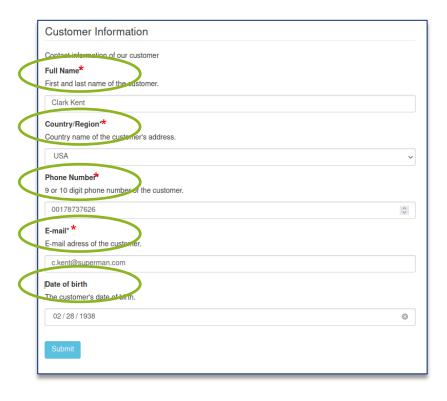
```
02 / 28 / 1938
```

Submit

CustomerInformationKentC.json

```
{

  "Full Name": "Clark Kent",
  "Country/Region": "USA",
  "Phone number": 00178737626,
  "E-mail": "c.kent@superman.com",
  "Date of birth": "1938-02-28"

}
```
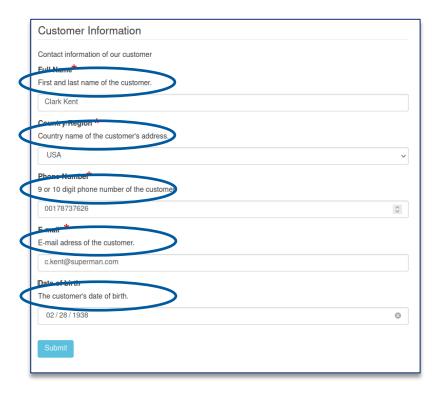
Controlled data submission – the form

Customer Information

Customer Information

Contact information of our customer

Full Name*
First and last name of the customer.

> Clark Kent

Country/Region*
Country name of the customer's address.

> USA

Phone Number*
9 or 10 digit phone number of the customer.

> 00178737626

E-mail *
E-mail adress of the customer.

> c.kent@superman.com

Date of birth
The customer's date of birth.

> 02 / 28 / 1938

[Submit]

CustomerInformationKentC.json

```
{

  "Full Name": "Clark Kent",
  "Country/Region": "USA",
  "Phone number": 00178737626,
  "E-mail": "c.kent@superman.com",
  "Date of birth": "1938-02-28"

}
```

Customer Information

Customer Information

Contact information of our customer

Full Name *
First and last name of the customer.

> Clark Kent

Country/Region *
Country name of the customer's address.

> USA

Phone Number *
9 or 10 digit phone number of the customer

> 00178737626

E-mail *
E-mail adress of the customer.

> c.kent@superman.com

Date of birth
The customer's date of birth.

> 02 / 28 / 1938

Submit

CustomerInformationKentC.json

```
{

  "Full Name": "Clark Kent",
  "Country/Region": "USA",
  "Phone number": 00178737626,
  "E-mail": "c.kent@superman.com",
  "Date of birth": "1938-02-28"

}
```

Benefits of schemas

A metadata schema is – basically – a set of **conventions or contraints.** [1]

Schemas are expressed in formal languages like XML, JSON or else
so that **(meta)data can be parsed and validated automatically according to the schema.** [2]

[1] https://www.merriam-webster.com/dictionary/schema, https://www.merriam-webster.com/dictionary/protocol
[2] https://gitlab.hzdr.de/hmc/hmc/cct-7-semantics/hmc-glossary-initialization/-/blob/master/terms/schema.yaml (HMC CCT7, not yet ratified)
Image: Child plays with wooden shape sorter toy, https://unsplash.com/photos/ehaO7XywMGM

XML Schemas (.xsd) are written in XML and used to describe & syntactically validate the structure of XML documents or (meta)data records. [1]

The JSON Schema vocabulary is used to describe & syntactically validate the structure of JSON (meta)data records. [2]

[1] "XML Schema Tutorial". © 1999-2022. Refsnes Data, W3Schools. https: //www.w3schools.com/xml/schema_intro.asp
[2] "Understanding JSON Schema. The basics", © Copyright 2013-2016 Michael Droettboom, Space Telescope Science Institute; Last updated on Feb 07, 2022. https://json-schema.org/understanding-json-schema/basics.html

We will **focus on writing JSON Schema** in our next hands-on task.

"$schema" vocabulary example

- JSON Schema version in $schema

- list of required properties

- one required property

- one optional property

- data type constraints

- descriptions for the human reader

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "description": "In real life you would add a meaningful description here.",
  "type": "object",
  "required": [
    "superhero"
  ],
  "properties": {
    "superhero": {
      "description": "A mandatory string property.",
      "type": "string"
    },
    "power": {
      "description": "An optional numeric property.",
      "type": "integer"
    }
  }
}
```

"$schema" vocabulary example

- list of required properties

- one required property

- one optional property

- data type constraints

- descriptions for the human reader

```json
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "description": "In real life you would add a meaningful description here.",
  "type": "object",
  "required": [
    "superhero"
  ],
  "properties": {
    "superhero": {
      "description": "A mandatory string property.",
      "type": "string"
    },
    "power": {
      "description": "An optional numeric property.",
      "type": "integer"
    }
  }
}
```

"$schema"

- list of required properties
- one required property
- one optional property
- data type constraints
- descriptions for the human reader

```json
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "description": "In real life you would add a meaningful description here.",
  "type": "object",
  "required": [
    "superhero"
  ],
  "properties": {
    "superhero": {
      "description": "A mandatory string property.",
      "type": "string"
    },
    "power": {
      "description": "An optional numeric property.",
      "type": "integer"
    }
  }
}
```

[1] www.json.org

"$schema"

- list of required properties

- one required property

- one optional property

- data type constraints

- descriptions for the human reader

```json
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "description": "In real life you would add a meaningful description here.",
  "type": "object",
  "required": [
    "superhero"
  ],
  "properties": {
    "superhero": {
      "description": "A mandatory string property.",
      "type": "string"
    },
    "power": {
      "description": "An optional numeric property.",
      "type": "integer"
    }
  }
}
```

[1] www.json.org

HELMHOLTZ

- list of required properties

- one required property

- one optional property

- data type constraints

- descriptions for the human reader

A JSON instance is syntactically valid, if it conforms to the definition described by the JSON schema.

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "description": "In real life you would add a meaningful description here.",
  "type": "object",
  "required": [
    "superhero"
  ],
  "properties": {
    "superhero": {
      "description": "A mandatory string property.",
      "type": "string"
    },
    "power": {
      "description": "An optional numeric property.",
      "type": "integer"
    }
  }
}
```

[1] www.json.org

Validate JSON record

A JSON instance is syntactically valid, if it conforms to the definition described by the JSON schema.

```
{
  "superhero": "String Hero"
}
```

```
{
  "superhero": 5
}
```

Customer Information