



Consuming REST services with Mendix

Outline

1. Case introduction
2. How to build a prototype using Mendix
3. Backgrounds on REST
4. Tips & Tricks
5. Q & A

Get external metadata into Mendix for selecting a song



eCard platform



REST



Spotify®

Music
Metadata

How to build this prototype using Mendix?

1. Extract REST endpoints needed from Spotify API documentation.
2. Figure out the messaging structure to search and collect metadata
3. Build the prototype in Mendix

Key concepts REST

Consumer

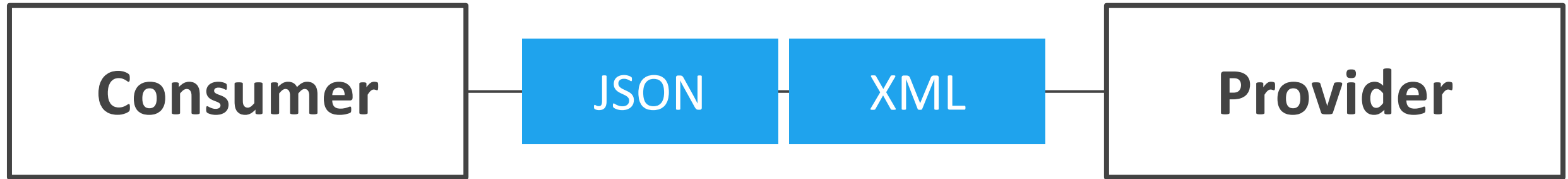
Provider

Key concepts REST

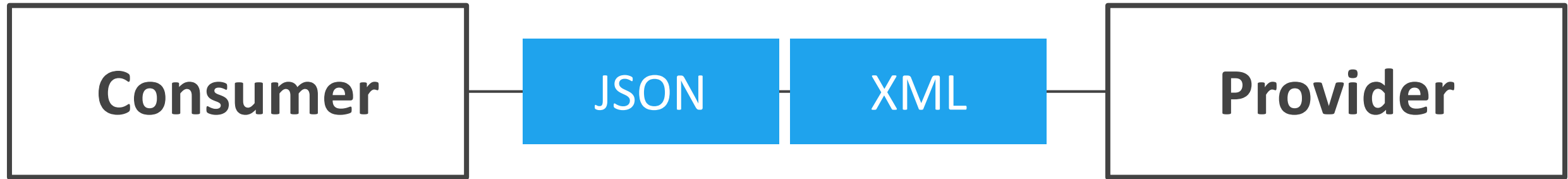


- Consumer initiates the call
- Provider receives it

Key concepts REST



Key concepts REST



Consumer consumes a service published by a provider

Key concepts REST

Consumer

JSON

XML

Provider

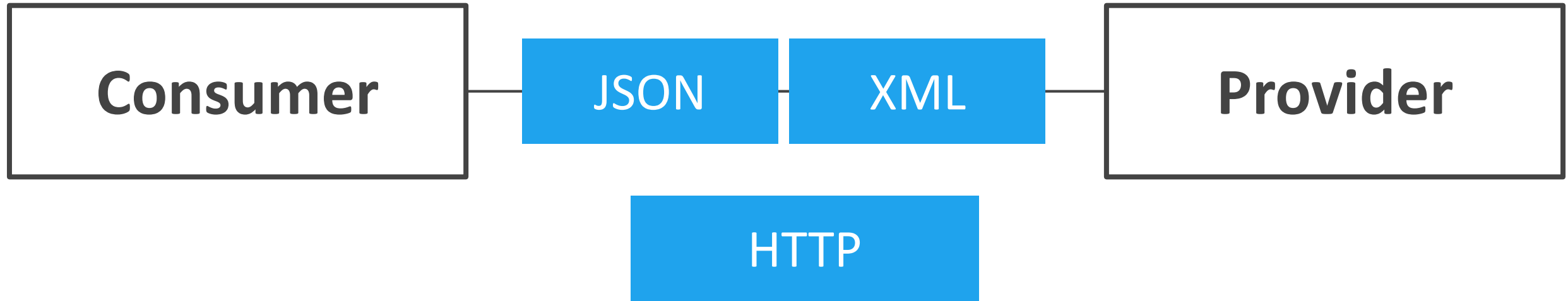
Representation in JSON:

```
1 {  
2   "ID": "1",  
3   "Name": "Joey Mendez",  
4   "Email": "j.mendez@gmail.com",  
5   "Country": "United Kingdom"  
6 }  
7
```

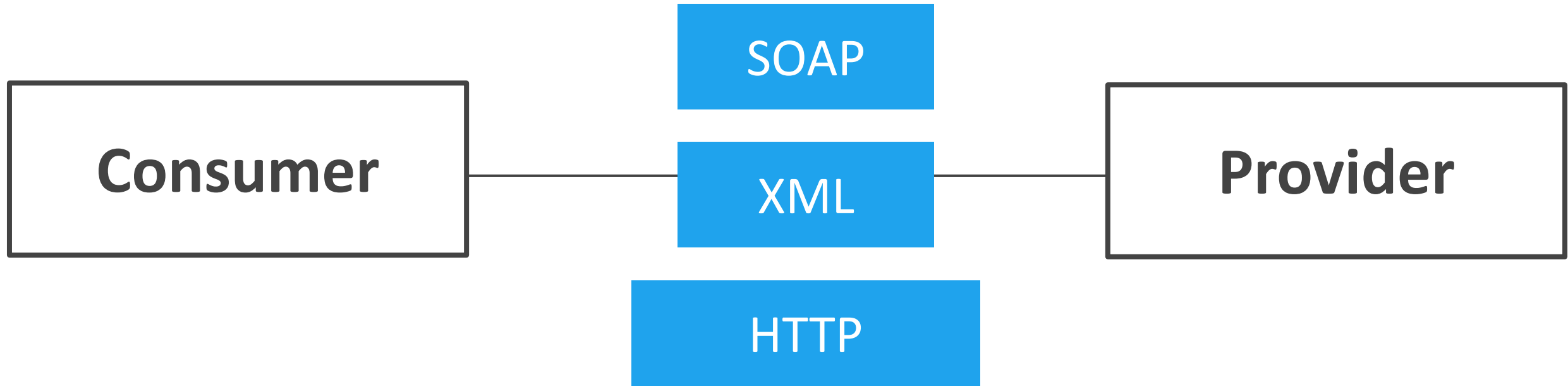
Representation in XML:

```
1 <Person>  
2  
3 <ID>1</ID>  
4  
5 <Name>Joey Mendez</Name>  
6  
7 <Email>j.mendez@gmail.com</Email>  
8  
9 <Country>United Kingdom</Country>  
10 </Person>  
11
```

Key concepts REST



Key concepts REST



HTTP-based integration methods in Mendix

<u>Mendix terminology:</u>	REST Service	Web Service	App Service	Odata Service
Data structures supported:	XML & JSON	SOAP (XML)	N/A	XML
Consume/Publish:	Consume only (publish using module)	Consume & publish	Consume & publish	Publish only
Distinctive characteristic / typical usage	Popular recently for public open APIs	Strict on typing, popular in enterprise contexts	Plug-and-play Mendix integration	Very easy to publish, suitable for BI/reporting purposes
Available from Mendix version:	6.6.0 natively, lower versions using module	2.x	5.x	5.18

Benefits of integrating using Mendix

JSON Structure 'SongSearcher.Spotify_JSON_structure'

General Documentation

JSON snippet

```
Format
```

```
{
  "tracks":{
    "href":"https://api.spotify.com/v1/search?query=jackson&offset=0&lin
    "items":[
      {
        "album":{
          "album_type":"album",
          "available_markets":[
            "AR",
            "AU",
            "AT",
            "BE",
            "BO",
            "BR",
            "BG",
            "CA",
            "CL",
```

Structure

Filter

Expand all Collapse all Refresh

Name	Value	Primitive ...	Occurr...	Null...	Custom Name
external urls			0.1		External urls 2
spotify	"https://ope...	String	0.1		Spotify
href	"https://api...	String	0.1		Href
id	"3cR4rhS2h...	String	0.1		id
name	"Leslie Odo...	String	0.1		Name
type	"artist"	String	0.1		type
uri	"spotify.artis...	String	0.1		Uri
available markets			0.1		Available_market
(Wrapper)			0.*		Wrapper 2
(Value)	"AR"	String	0.1		Value
disc number	1	Integer	0.1		Disc number
duration ms	236737	Integer	0.1		Duration ms
explicit	false	Boolean	0.1		Explicit
external ids			0.1		External ids
isrc	"USAT2150...	String	0.1		Isrc
external urls			0.1		External urls 3
spotify	"https://ope...	String	0.1		Spotify
href	"https://api...	String	0.1		Href
id	"4TTV7Ecf...	String	0.1		id
name	"Alexander...	String	0.1		Name
popularity	73	Integer	0.1		Popularity
preview url	"https://p.sc...	String	0.1		Preview url
track number	1	Integer	0.1		Track number
type	"track"	String	0.1		type
uri	"spotify.trac...	String	0.1		Uri
limit	20	Integer	0.1		Limit
next	"https://api...	String	0.1		Next
offset	0	Integer	0.1		Offset
previous	null	Unknown	0.1		Previous
total	60502	Integer	0.1		Total

OK Cancel

Drag parameter entity here (optional)

