FHIR Conversion Made Easy

Open-source FUME's Growing Impact on FHIR Implementations

Daniel Mechanik | Outburn LTD.

HL7 FHIR DevDays 2024 | Minneapolis, MN | June 10-13, 2024 | @HL7 | @FirelyTeam | #fhirdevdays | www.devdays.com

ORGANIZED BY firely | HL7 International

Who am I?

Daniel Mechanik

Founder & CTO at Outburn LTD

Data & Integration Architect

FHIR Evangelist

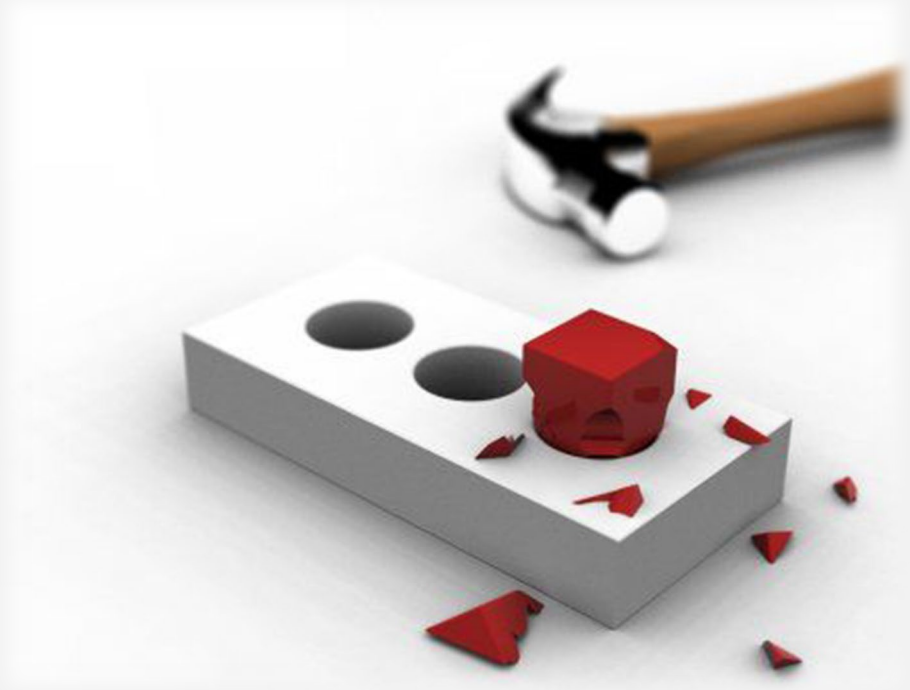
16+ years in Health Tech

The Challenge

Conversion of data to FHIR is not as straight-forward as it should be.

It’s not just the data structure…

In-fact, **the structure is the easy part**!



The **Real** Challenge

The Contents

- Correct use of URI's/URL's (Extensions, Codings, Identifiers)
- Understanding profiles, bindings and slices
- Performing terminology conversions
- Each use case/profile requires **many** fixed values
  (usually hard-coded)

The **Real** Challenge

Working with a resource repository (without destroying the data…)

In most scenarios, we need to query the server to decide:

- What CRUD operations to perform on each resource?
- Reference an existing resource or create a new one?

➔ Conversion of the **same source record** will have **different results**

- With different FHIR servers
- At different points in time

It's NOT a "stateless" process!

The **Real** Challenge

Existing tools (plug-ins & SDK's) are platform/language specific

- Can't reuse across organizations – even when implementing the same exact thing
- Vendor lock-in - replacing tools means re-doing the work
- Must be a programmer
  - Good data & FHIR experts may be bad programmers (or not at all)
  - Good programmers may be bad data & FHIR experts (or not at all)
    - Many FHIR decision-makings SHOULD NOT be left to programmers!
  - Programmers are expensive

The **Real** Challenge

Validation

The extent of validation is always limited

- Varies greatly between tools
- Usually just the base FHIR schema
- Code validation - limited (or none)
- Cardinality – only base, never on slices

➔ Problems only surface later (in production)

☹ Poor data quality

The **Real** Challenge

Profiles

- Conformance to profiles is entirely up to the implementor
- It’s a manual effort - and requires expertise

  (Weird, isn’t it? Aren't profiles machine-readable?)
- All fixed values (URI’s, codes etc.) need to be copy-pasted into the code
  - Prone to errors
  - Hard to maintain in the long run
  - Painful if a URI needs to be changed

The Other Side of the Challenge

Existing FHIR data is also not easy to work with

- Simple questions may require chains of API calls
  - May be too demanding for some client applications
- Extracting simple data from FHIR resources requires expertise
- Legacy systems may not be able to work directly with FHIR API's / Resources
- Need to resolve references to see the whole picture

➔ We often need a "reverse façade"

The Other Side of the Challenge

Not all FHIR servers are the same

- Clients can only rely on widely supported capabilities
  - Need to compensate for this somehow
- There is no vendor-agnostic method to implement missing capabilities on the server
  - OperationDefinition & SearchParameter resources are not executable

The Other Side of the Challenge

FHIR-to-FHIR Conversions

- QuestionnaireResponse ➔ Observation
- FHIR Version A ➔ FHIR Version B
- Profile A ➔ Profile B
- Patch (adding data to existing resources)

The Vision

A uniform method for FHIR transformations, that is:

- Production ready & performant
- Easy to author & maintain - even for non-programmers
- Platform agnostic – transformations called through a RESTful API
- Declarative – more focus on the “what”, less on the “how”
- Maximizing the use of FHIR assets
  - Supporting profiles automatically
  - Allows sharing & reuse through StructureMap, ConceptMap resources
- Extensible & Modular
- **Open-source**

Options We Ruled-out

The FHIR Mapping Language

- Not production ready
  - More of an academic experiment
  - Limited function library
  - No supporting tools – currently only the HL7 Java Validator
- Not easy to author – syntax is not intuitive enough
- Not declarative – procedural in its nature
- Not maximizing the use of FHIR profiles
  - Doesn’t support assignment of values into slices
  - Fixed values need to be hard coded
- Not Extensible

Options We Ruled-out

Using FHIR Shorthand as-is (Instance declarations)

- The FSH library is built for profiling
  - Performance is not tailored towards production use
- Only fixed values can be used
  - Syntax is not defined for dynamic inputs

FUME – The **FHIR-Utilized Mapping Engine**

An open-source "converter API factory"

- A Mapping Language inspired by FHIR Shorthand: https://fume.health/docs/flash
- Data extraction & manipulation using JSONata (IBM)
- Profiles, slices & extensions can be used explicitly
  - Fixed values injected automatically
- Validation on-the-go
- Inline FHIR API calls & terminology conversions
- Mappings as StructureMaps - can be shared and reused
- Native support for JSON, CSV and HL7v2
- Easily extended with additional functionalities

Invoking transformations:

- Through the API:
  - POST to a specific mapping endpoint
  - POST source data and an expression
- From within other mappings
  - Just call you mappings – they are functions!

GitHub: https://github.com/Outburn-IL/fume-community

Public interactive playground: https://try.fume.health/

Example – Blood Pressure Profile

Source

```
{
  "patient_mrn": "PP875023984",
  "doctor_id": "1-820958",
  "date": "2024-06-07T18:36:46.006Z",
  "systolic": 110,
  "diastolic": 78
}
```

Example – Blood Pressure Profile

Source

```
{
  "patient_mrn": "PP875023984",
  "doctor_id": "1-820958",
  "date": "2024-06-07T18:36:46.006Z",
  "systolic": 110,
  "diastolic": 78
}
```

Mapping

```
InstanceOf: bp
* status = 'final'
* effectiveDateTime = date
* subject
  * reference = $literal('Patient', {'identifier': patient_mrn})
* performer
  * reference = $literal('Practitioner', {'identifier': doctor_id})
* component[SystolicBP].value
  * value = systolic
* component[DiastolicBP].value
  * value = diastolic
```

Example – Blood Pressure Profile

Source

```
{
  "patient_mrn": "PP875023984",
  "doctor_id": "1-820958",
  "date": "2024-06-07T18:36:46.006Z",
  "systolic": 110,
  "diastolic": 78
}
```

Mapping

```
InstanceOf: bp
* status = 'final'
* effectiveDateTime = date
* subject
  * reference = $literal('Patient', {'identifier': patient_mrn})
* performer
  * reference = $literal('Practitioner', {'identifier': doctor_id})
* component[SystolicBP].value
  * value = systolic
* component[DiastolicBP].value
  * value = diastolic
```

Result:

```
{
  "resourceType": "Observation",
  "meta": {
    "profile": [
      "http://hl7.org/fhir/StructureDefinition/bp"
    ]
  },
  "category": [
    {
      "coding": [
        {
          "system": "http://terminology.hl7.org/CodeSystem/observation-category",
          "code": "vital-signs"
        }
      ]
    }
  ],
  "code": {
    "coding": [
      {
        "system": "http://loinc.org",
        "code": "85354-9"
      }
    ]
```

Example – Blood Pressure Profile

Source

```
{
  "patient_mrn": "PP875023984",
  "doctor_id": "1-820958",
  "date": "2024-06-07T18:36:46.006Z",
  "systolic": 110,
  "diastolic": 78
}
```

Mapping

```
InstanceOf: bp
* status = 'final'
* effectiveDateTime = date
* subject
  * reference = $literal('Patient', {'identifier': patient_mrn})
* performer
  * reference = $literal('Practitioner', {'identifier': doctor_id})
* component[SystolicBP].value
  * value = systolic
* component[DiastolicBP].value
  * value = diastolic
```

Result:

```
},
"component": [
  {
    "code": {
      "coding": [
        {
          "system": "http://loinc.org",
          "code": "8480-6"
        }
      ]
    },
    "valueQuantity": {
      "system": "http://unitsofmeasure.org",
      "code": "mm[Hg]",
      "value": 110,
      "unit": "millimeter of mercury"
    }
  },
  {
    "code": {
      "coding": [
        {
          "system": "http://loinc.org",
          "code": "8462-4"
```

Example – Blood Pressure Profile

Source

```
{
  "patient_mrn": "PP875023984",
  "doctor_id": "1-820958",
  "date": "2024-06-07T18:36:46.006Z",
  "systolic": 110,
  "diastolic": 78
}
```

Mapping

```
InstanceOf: bp
* status = 'final'
* effectiveDateTime = date
* subject
  * reference = $literal('Patient', {'identifier': patient_mrn})
* performer
  * reference = $literal('Practitioner', {'identifier': doctor_id})
* component[SystolicBP].value
  * value = systolic
* component[DiastolicBP].value
  * value = diastolic
```

Result:


```
            "system": "http://loinc.org",
            "code": "8462-4"
          }
        ]
      },
      "valueQuantity": {
        "system": "http://unitsofmeasure.org",
        "code": "mm[Hg]",
        "value": 78,
        "unit": "millimeter of mercury"
      }
    }
  ],
  "status": "final",
  "effectiveDateTime": "2024-06-07T18:36:46.006Z",
  "subject": {
    "reference": "Patient/38bf312b-9f90-4302-ab8f-ff2ff6732c34"
  },
  "performer": [
    {
      "reference": "Practitioner/cc829d28-3b32-43df-af57-e72035d98e18"
    }
  ]
}
```

Comments, Functions, Aliases, Logics & Code Translation

Source

```
{
  "status": "active",
  "ssn": "123-45-6789",
  "first_name": "Jessica",
  "last_name": "Rabbit",
  "birth_date": "06/22/1988",
  "sex": "F"
}
```

Comments, Functions, Aliases, Logics & Code Translation

Source

```
{
  "status": "active",
  "ssn": "123-45-6789",
  "first_name": "Jessica",
  "last_name": "Rabbit",
  "birth_date": "06/22/1988",
  "sex": "F"
}
```

Mapping

```
// define local fnction
$DateUStoFhir := function($date){
  $fromMillis($toMillis($date,'[M01]/[D01]/[Y]'))
};

InstanceOf: Patient
* identifier
  * system = $ssn // using global alias
  * value = ssn
* active = (status='active') // boolean logic
* name
  * given = first_name
  * family = last_name
  // calling our function:
* birthDate = $DateUStoFhir(birth_date)
// translating codes:
* gender = $translate(sex, 'gender')
```

Comments, Functions, Aliases, Logics & Code Translation

Source

```
{
  "status": "active",
  "ssn": "123-45-6789",
  "first_name": "Jessica",
  "last_name": "Rabbit",
  "birth_date": "06/22/1988",
  "sex": "F"
}
```

Mapping

```
// define local fnction
$DateUStoFhir := function($date){
  $fromMillis($toMillis($date,'[M01]/[D01]/[Y]'))
};

InstanceOf: Patient
* identifier
  * system = $ssn // using global alias
  * value = ssn
* active = (status='active') // boolean logic
* name
  * given = first_name
  * family = last_name
  // calling our function:
* birthDate = $DateUStoFhir(birth_date)
// translating codes:
* gender = $translate(sex, 'gender')
```

Result:

```
{
  "resourceType": "Patient",
  "identifier": [
    {
      "system": "http://hl7.org/fhir/sid/us-ssn",
      "value": "123-45-6789"
    }
  ],
  "active": true,
  "name": [
    {
      "given": [
        "Jessica"
      ],
      "family": "Rabbit"
    }
  ],
  "birthDate": "1988-06-22",
  "gender": "female"
}
```

String Manipulation, Extensions, Context & Arrays

Source

```
{
  "id": "75023983",
  "first_name": "Jessica",
  "last_name": "Rabbit",
  "address": {
    "city_code": "3000",
    "lat": 28.3519592,
    "long": -81.417283
  },
  "phones": ["+972-958-6438","+972-678-6364"]
}
```

String Manipulation, Extensions, Context & Arrays

Source

```
{
  "id": "75023983",
  "first_name": "Jessica",
  "last_name": "Rabbit",
  "address": {
    "city_code": "3000",
    "lat": 28.3519592,
    "long": -81.417283
  },
  "phones": ["+972-958-6438","+972-678-6364"]
}
```

Mapping

```
InstanceOf: il-core-patient
* identifier[il-id].value = $pad(id,-9,'0')
* name.text = first_name & ' ' & last_name
* gender.extension[data-absent-reason].value = 'unknown'
* (address).address
  * extension[geolocation]
    * extension[latitude].value = lat
    * extension[longitude].value = long
  * city.extension[cityCode].value.coding
    * system = $citySymbol
    * code = city_code
* (phones).telecom
  * system = 'phone'
  * value = $
```

String Manipulation, Extensions, Context & Arrays

Source

```
{
  "id": "75023983",
  "first_name": "Jessica",
  "last_name": "Rabbit",
  "address": {
    "city_code": "3000",
    "lat": 28.3519592,
    "long": -81.417283
  },
  "phones": ["+972-958-6438","+972-678-6364"]
}
```

Mapping

```
InstanceOf: il-core-patient
* identifier[il-id].value = $pad(id,-9,'0')
* name.text = first_name & ' ' & last_name
* gender.extension[data-absent-reason].value = 'unknown'
* (address).address
  * extension[geolocation]
    * extension[latitude].value = lat
    * extension[longitude].value = long
  * city.extension[cityCode].value.coding
    * system = $citySymbol
    * code = city_code
* (phones).telecom
  * system = 'phone'
  * value = $
```

Result:

```
{
  "resourceType": "Patient",
  "meta": {…
  },
  "identifier": [
    {
      "system": "http://fhir.health.gov.il/identifier/il-national-id",
      "value": "075023983"
    }
  ],
  "name": [
    {
      "text": "Jessica Rabbit"
    }
  ],
  "_gender": {
    "extension": [
      {
        "url": "http://hl7.org/fhir/StructureDefinition/data-absent-reason",
        "valueCode": "unknown"
      }
    ]
  },
  "address": [
```

String Manipulation, Extensions, Context & Arrays

Source

```
{
  "id": "75023983",
  "first_name": "Jessica",
  "last_name": "Rabbit",
  "address": {
    "city_code": "3000",
    "lat": 28.3519592,
    "long": -81.417283
  },
  "phones": ["+972-958-6438","+972-678-6364"]
}
```

Mapping

```
InstanceOf: il-core-patient
* identifier[il-id].value = $pad(id,-9,'0')
* name.text = first_name & ' ' & last_name
* gender.extension[data-absent-reason].value = 'unknown'
* (address).address
  * extension[geolocation]
    * extension[latitude].value = lat
    * extension[longitude].value = long
  * city.extension[cityCode].value.coding
    * system = $citySymbol
    * code = city_code
* (phones).telecom
  * system = 'phone'
  * value = $
```

Result:

```
"address": [
  {
    "extension": [
      {
        "extension": [
          {
            "url": "latitude",
            "valueDecimal": 28.3519592
          },
          {
            "url": "longitude",
            "valueDecimal": -81.417283
          }
        ],
        "url": "http://hl7.org/fhir/StructureDefinition/geolocation"
      }
    ],
    "_city": {
      "extension": [
        {
          "url": "http://fhir.health.gov.il/StructureDefinition/ext-city-code",
          "valueCodeableConcept": {
            "coding": [
              {
```

String Manipulation, Extensions, Context & Arrays

Source

```
{
  "id": "75023983",
  "first_name": "Jessica",
  "last_name": "Rabbit",
  "address": {
    "city_code": "3000",
    "lat": 28.3519592,
    "long": -81.417283
  },
  "phones": ["+972-958-6438","+972-678-6364"]
}
```

Mapping

```
InstanceOf: il-core-patient
* identifier[il-id].value = $pad(id,-9,'0')
* name.text = first_name & ' ' & last_name
* gender.extension[data-absent-reason].value = 'unknown'
* (address).address
  * extension[geolocation]
    * extension[latitude].value = lat
    * extension[longitude].value = long
  * city.extension[cityCode].value.coding
    * system = $citySymbol
    * code = city_code
* (phones).telecom
  * system = 'phone'
  * value = $
```

Result:

```
            "valueCodeableConcept": {
              "coding": [
                {
                  "system": "http://fhir.health.gov.il/cs/city-symbol",
                  "code": "3000"
                }
              ]
            }
          }
        ]
      }
    }
  ],
  "telecom": [
    {
      "system": "phone",
      "value": "+972-958-6438"
    },
    {
      "system": "phone",
      "value": "+972-678-6364"
    }
  ]
}
```

Reverse Façade – Practitioner API

Input

```
{
  "license_num": "1-820958"
}
```

Reverse Façade – Practitioner API

Input

```
{
  "license_num": "1-820958"
}
```

Mapping

```
$search( // perform FHIR search
  'Practitioner',
  {
    'identifier': 'http://fume.health/identifier/practitioner-license|'
     & license_num
  }
).entry[0].resource.{ // transform result
  'license_num': $$.license_num,
  'first_name': name.given[0],
  'last_name': name.family,
  'status': active ? 'active' : 'inactive'
}
```

Reverse Façade – Practitioner API

Input

```
{
  "license_num": "1-820958"
}
```

Mapping

```
$search( // perform FHIR search
  'Practitioner',
  {
    'identifier': 'http://fume.health/identifier/practitioner-license|'
      & license_num
  }
).entry[0].resource.{ // transform result
  'license_num': $$.license_num,
  'first_name': name.given[0],
  'last_name': name.family,
  'status': active ? 'active' : 'inactive'
}
```

Result:

```
{
  "license_num": "1-820958",
  "first_name": "Doctor",
  "last_name": "Dolittle",
  "status": "active"
}
```

Server Resource Count

Mapping

```
$capabilities()
.rest
.resource
.type // array of resource types supported by server
{
  $: // set a key for each resource type
  $search($, {'_summary': 'count'}) // perform a count search
  .(
    // return only if greater than 25
    total > 25 ? total
  )
}
```

Server Resource Count

Mapping

```
$capabilities()
.rest
.resource
.type // array of resource types supported by server
{
  $: // set a key for each resource type
  $search($, {'_summary': 'count'}) // perform a count search
  .(
    // return only if greater than 25
    total > 25 ? total
  )
}
```

Result

```
{
  "Binary": 165,
  "Bundle": 56,
  "CodeSystem": 27,
  "ConceptMap": 29,
  "Condition": 43,
  "Encounter": 137,
  "Group": 82,
  "List": 32,
  "MedicationRequest": 26,
  "Observation": 267,
  "Organization": 38,
  "Patient": 567,
  "Practitioner": 39,
  "RelatedPerson": 122,
  "StructureDefinition": 150,
  "StructureMap": 269,
  "Task": 49,
  "ValueSet": 116
}
```

Large-Scale Real-Life Test Cases

- Ministry of Health Israel
  - Populating national FHIR server with public-health data received in diverse formats
  - Extending support for XML inputs
- Rambam Medical Center
  - Populating FHIR server with their custom-EMR data
    - All according to custom profiles, derived from IL-Core profiles
  - Extracting & Anonymizing FHIR data for research
- Laniado Medical Center
  - Populating FHIR server with their custom-EMR data
    - All according to custom profiles, derived from IL-Core profiles

Other Interesting Implementations (that we know of)

- "Reverse façade" – Exposing FHIR data as legacy API's
- FHIR Interceptor
  - Implementing security policies (e.g. masking attributes)
  - Adding capabilities to the server (search params, operations)
    (In a shareable & executable format)
  - Executing external interfaces using API calls
- Data Quality Assurance
  - Periodically fetch bulks of resources, and:
    - Check DQA rules
    - Perform statistics
    - Perform in-depth validation (with HL7's Java Validator)

Impact of FUME on FHIR Implementations

Encourages organizations to create their own profiles (and publish them)

- All URI's and fixed values are managed in the profile
  - Single place to define, maintain and make changes
- No hard-coded strings in the transformation logic
  - Transformations deal only with data mapping and not with fixed values
- Easy for them to learn FSH
  - Since the syntax is similar

➔ ☺ Improved data quality

Impact of FUME on FHIR Implementations

Data experts take full responsibility for data transformations

- Programmers only need to call the appropriate mapping API
- Legacy systems don’t need to learn FHIR
  - They can use a “reverse-façade” API generated for them
- Errors are easier to debug - and faster to fix

Impact of FUME on FHIR Implementations

Encourages creativity

- Easy to chain together FHIR API calls
- Easy to manipulate data
- Easy to extend functionality as needed
  - Using FUME's syntax
  - Using any JavaScript library
  - Using any external API

➔ People are using FUME in ways we did not expect!

Impact of FUME on FHIR Implementations

Opens-up the market for more FHIR Server vendors

- Missing capabilities can be implemented without creating vendor lock-in
- Less stress on decision makers to choose the “right” vendor

Useful Links

- Video Tutorials
- Mapping Language Documentation
- The Interactive Sandbox: try.fume.health
- GitHub Repo: github.com/Outburn-IL/fume-community
  - Give us a star!

Contact

outburn.co.il

daniel@outburn.co.il

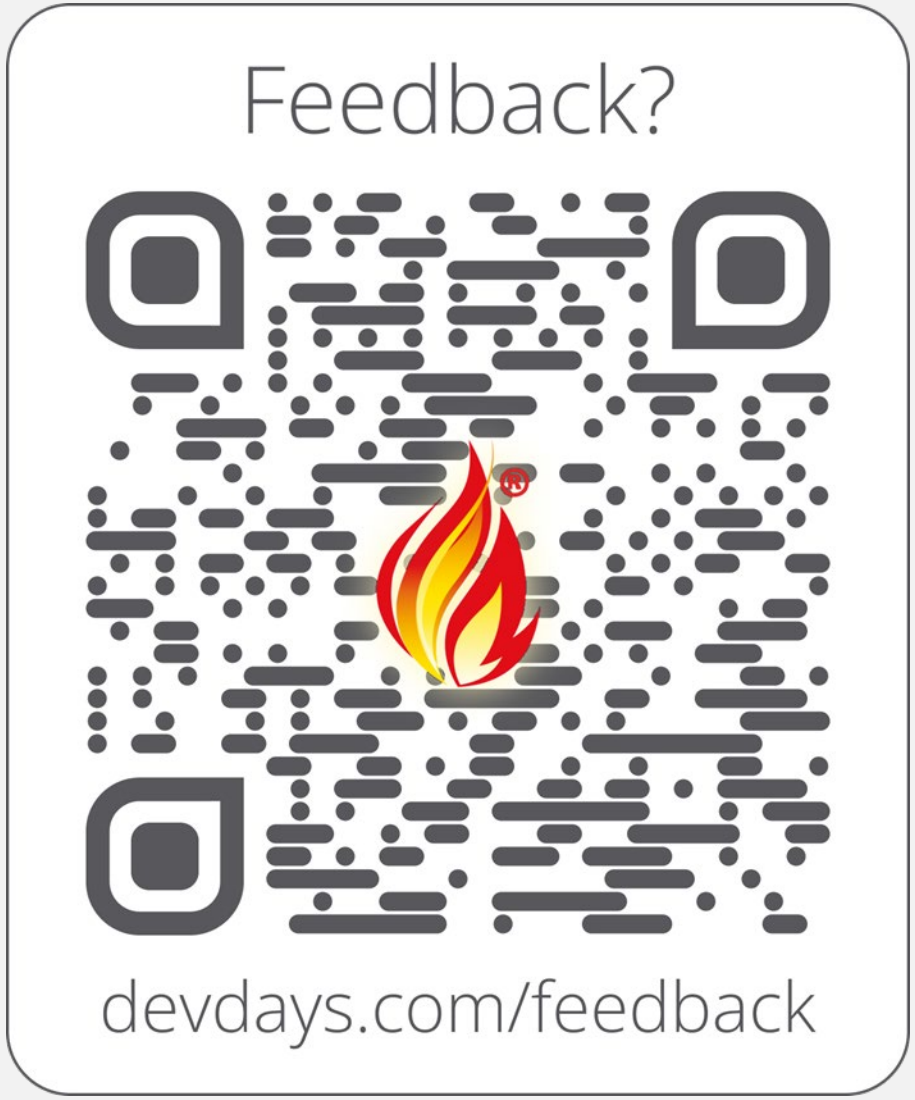
linkedin.com/in/danielmechanik

Thank you!

Q&A

Feedback

https://www.devdays.com/feedback/



ORGANIZED BY