SmartNSF

100% smart and in Color

brought to you by

Christian Güdemann
OpenNTF Chairman

CTO@WebGate Consulting AG

Martin Jinoch
freelance SW engineer

Agenda and Co

- You ask Why..?
- Lets Start!
- Strategies for Views and Docs
- It‘s Groovy Baby! – 7 Events to go
- Documentation? What a question... Lets Swagger

You ask Why???

- It all began with the Domino, Detail and Devil Problem aka:

BUSINESS LOGIC is EVERYWHERE

The difference between …

- Business Logic
  - Your rules and dependency about your business like:
    - If you want to register a new customer, please provide the VAT Number
    - No Company without Contact
  - Business Logic is independent from the client
- Application behaviour
  - Covers questions about “How can I register a new customer”
  - Type-a-head for numbers and values
  - Is client agnostic (Web Client acts different then Mobile Client)

The Inspiration

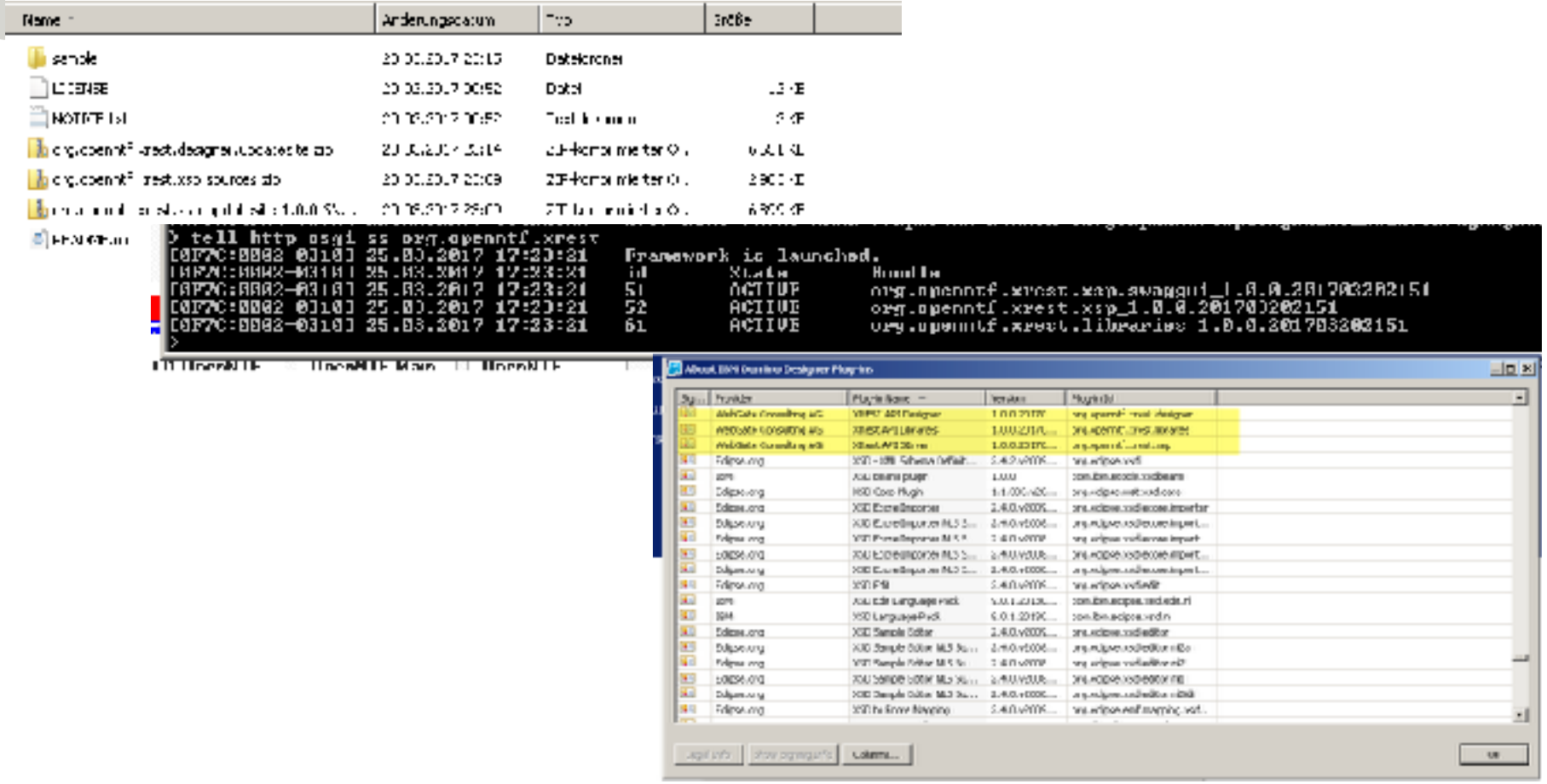
How can I convert a simple NSF into a micro-service within minutes?! (To protect the business logic)

```
app.get('/', function (req, res) {
            res.send('Hello World!')
})

app.post('/', function (req, res) {
            res.send('Got a POST request')
})

app.put('/user', function (req, res) {
res.send('Got a PUT request at /user')
})

app.delete('/user', function (req, res) {
res.send('Got a DELETE request at /user') })
```

```
router.GET('topics') {
            strategy(DOCUMENTS_BY_VIEW) {
                        viewName('($All)')
            }
            mapJson 'id', json:'id', type:'STRING',
isformula:true,
formula:'@DocumentUniqueID'
            mapJson "date", json:'date',type:'DATETIME',
                                    isformula:true,
formula:'@Created'
            mapJson "Subject", json:'topic', type:'STRING'
            mapJson "author", json:'author', type:'STRING',

isformula:true,formula:'@Name([CN]; From)'
}
```

Let’s go

https://openntf.org/main.nsf/project.xsp?r=project/SmartNSF



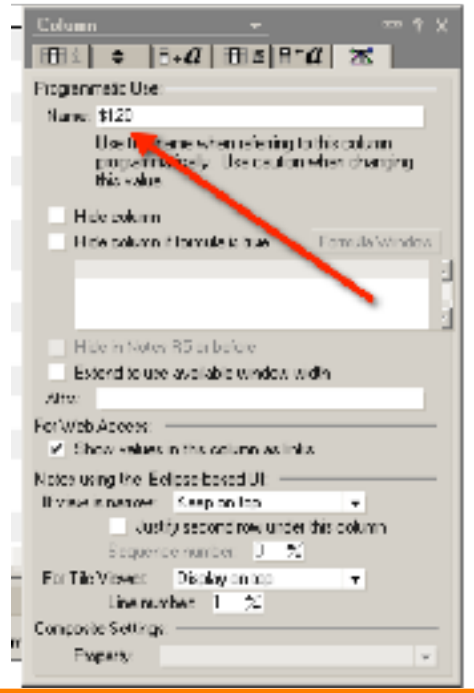
Some words about REST

- The power of the „methods”:
  - GET -> Give me (one or more objects)
  - POST -> Create a new object
  - PUT -> Replace the object
  - UPDATE -> Update the properties that we submit
  - DELETE -> Delete the object

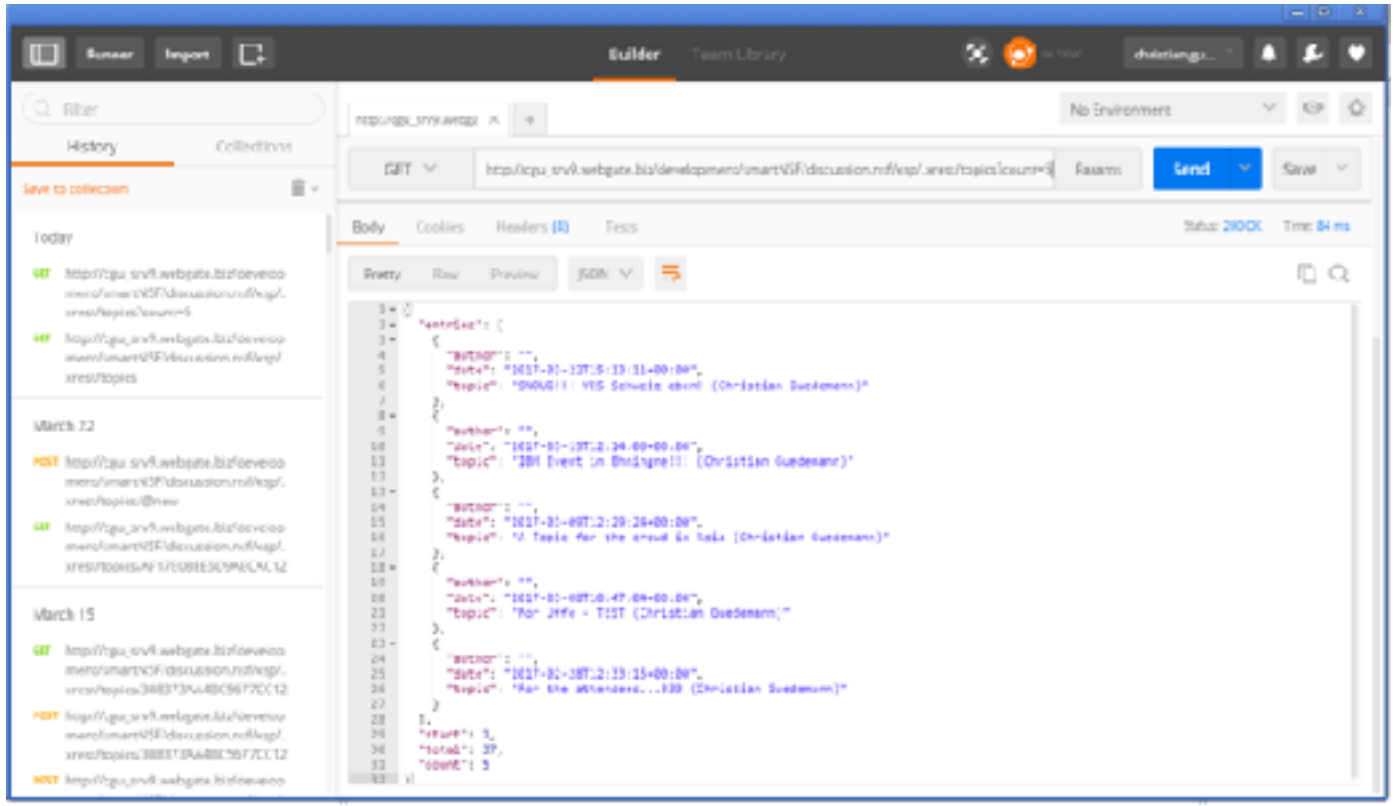
- We do only simple GET and POST (for today only)

Show me the view (not the money) but fast!

```
router.GET(‘topics’) {
  strategy(VIEWENTRIES_PAGED) {
    viewName('(ByDate)')
  }
  mapJson '\$120', json:'topic', type:’STRING'
  mapJson 'author', json:'author', type:’STRING'
  mapJson 'date', json:’date', type:'DATETIME'
}
```



Try it…


Runner
Import
Builder
Team Library
Filter
History
Collections
Save to collection
Today
March 22
March 15
No Environment
GET
Params
Send
Save
Body
Cookies
Headers (8)
Tests
Status: 200 OK
Time: 84 ms
Pretty
Raw
Preview
JSON

Route for a document

```
router.GET('topics/{id}') {
  strategy(DOCUMENT_BY_UNID) {
    keyVariableName("id")
  }
  mapJson "date", json:’date',type:'STRING', isformula:true,
                    formula:'@Text(@Created)'
  mapJson "Subject", json:'topic', type:'STRING'
  mapJson "author", json:'author', type:’STRING', isformula:true,
                    formula:'@Name([CN]; From)'
  mapJson "body", json:'content', type:'MIME'
  mapJson "categories", json:'categories', type:'ARRAY_OF_STRING'
}
```

Strategies

- DOCUMENT_BY_UNID
- DOCUMENT_FROM_VIEW_BY_KEY

- DOCUMENTS_BY_SEARCH_FT_PAGED
- DOCUMENTS_BY_FORMULA_PAGED
- DOCUMENTS_BY_VIEW_PAGED
- DOCUMENTS_FROM_VIEW_BY_KEY_PAGED

- VIEWENTRIES_PAGED
- VIEWENTRIES_BY_CATEGORY_PAGED

- ATTACHMENT

Note: All DOCUMENTS* and VIEWENTRIES* strategies also have non paged versions. Use with caution, these return the complete dataset at once.

Strategies

- All _PAGED Strategies
  - Using start and count as URL Parameters
  - Are designed to deliver a fast user experience
- All VIEWENTRIES Strategies
  - Are using a view as model
  - Do not have a direct access to the document
  - Are “<<<your word>>>” fast
- All DOCUMENT Strategies
  - Are slower then the VIEWENTRIES Strategies but….
  - Let you evaluate complex @Formulas for mapJson mapping

Power of events

- Every Route supports these events
  - VALIDATE
  - PRE_LOAD_DOCUMENT
  - POST_LOAD_DOCUMENT
  - POST_NEW_DOCUMENT
  - PRE_SAVE_DOCUMENT
  - POST_SAVE_DOCUMENT
  - PRE_SUBMIT

Groovy in event's context and Model

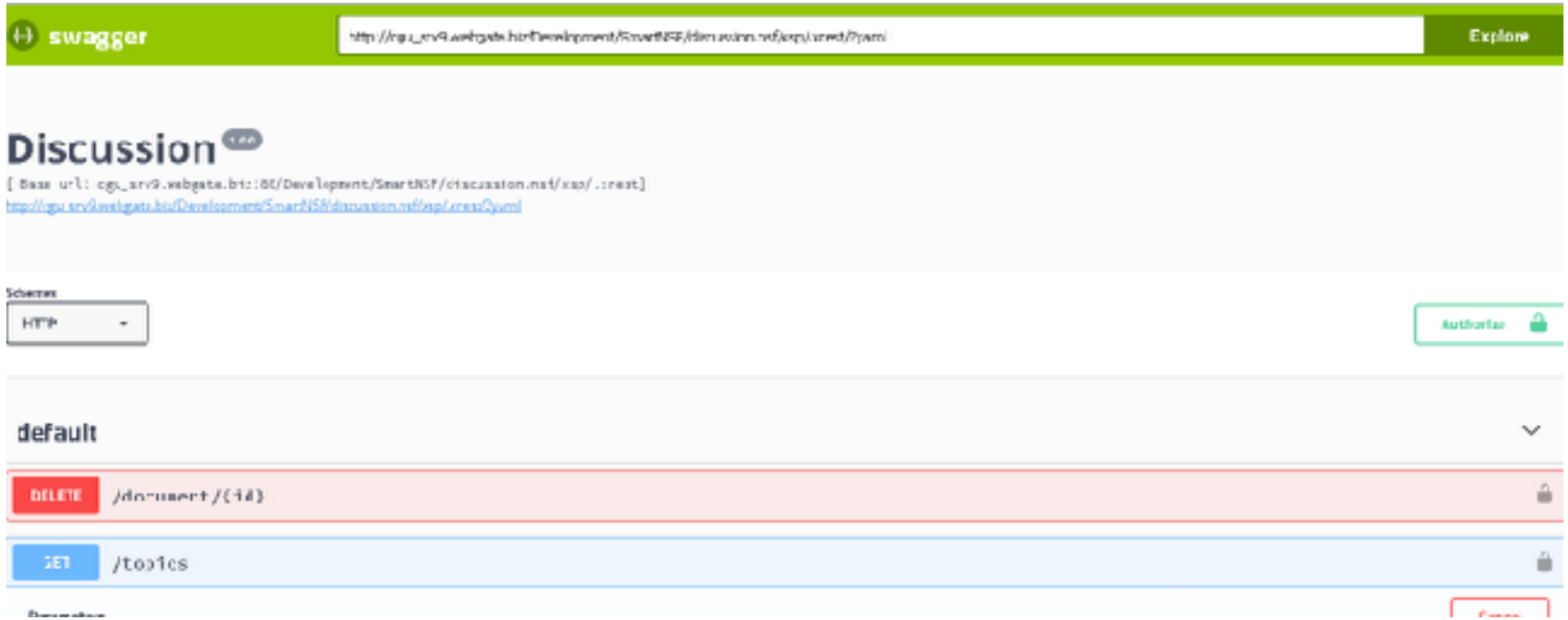
- Every Event has a closure
- In each closure the context and the document model are submitted

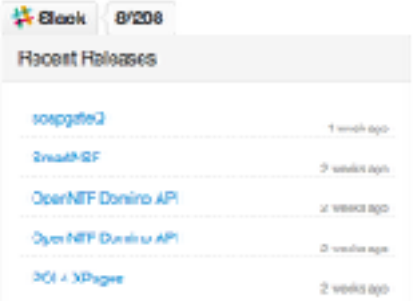
- Btw. the closure gives you access to the full Java Stack!

Open API Specification, Swagger….

- You wont believe, but your Code is

documented! - Built in!



Call to action

- Download the project
- Enter OpenNTS' Slack space and become part of #smartnsf