codecentric

REST APIs from Hell

- Wahre Geschichten über Misserfolge -

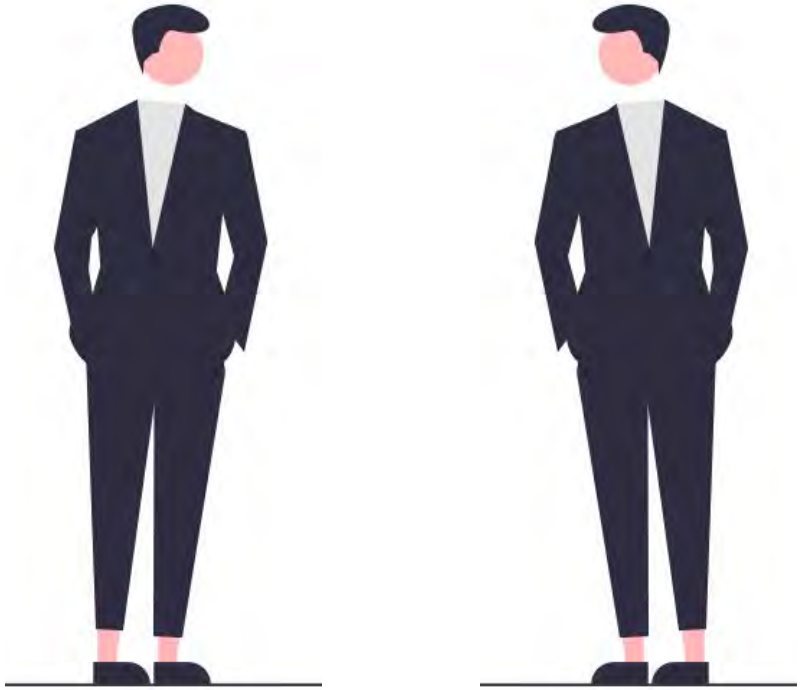
Disclaimer

Alle Geschichten aus dieser Präsentation sind tatsächlich so passiert. Dies soll jedoch niemanden bloßstellen. Keine der beteiligten Personen hat absichtlich oder böswillig so gehandelt.

Idee: Wir müssen unsere Abläufe automatisieren und dafür bauen wir eine mobile Anwendung!

“Aber wie kommen wir an die Daten aus dem ERP-System?”

Idee: Die zuständige Abteilung muss eine API für das ERP-System bereitstellen.



“Ihr müsst eine API für uns bereitstellen.”

“Wir brauchen dringend eine API.”

Herausforderungen

Unklare Anforderungen

Was die API machen soll und wer die API nutzen soll ist dem Entwicklungsteam nicht klar.

Spezifische Skills

Das Entwicklungsteam hat im schlechtesten Fall keinerlei Erfahrung mit modernen Entwicklungsmethoden und APIs.

Zeit- & Kostendruck

Der Zeit- und Kostendruck ist hoch, das heißt die API muss mit möglichst wenig Aufwand bereits gestern fertig sein.

Miri (sie/ihr)

Dr. Miriam Greis
Lead API Consultant
miriam.greis@codecentric.de
https://www.linkedin.com/in/miriam-greis

Azure DevOps

OPEN API INITIATIVE

Stoplight

CI/CD

NEWMAN

Portman

Vue.js

Jest

node

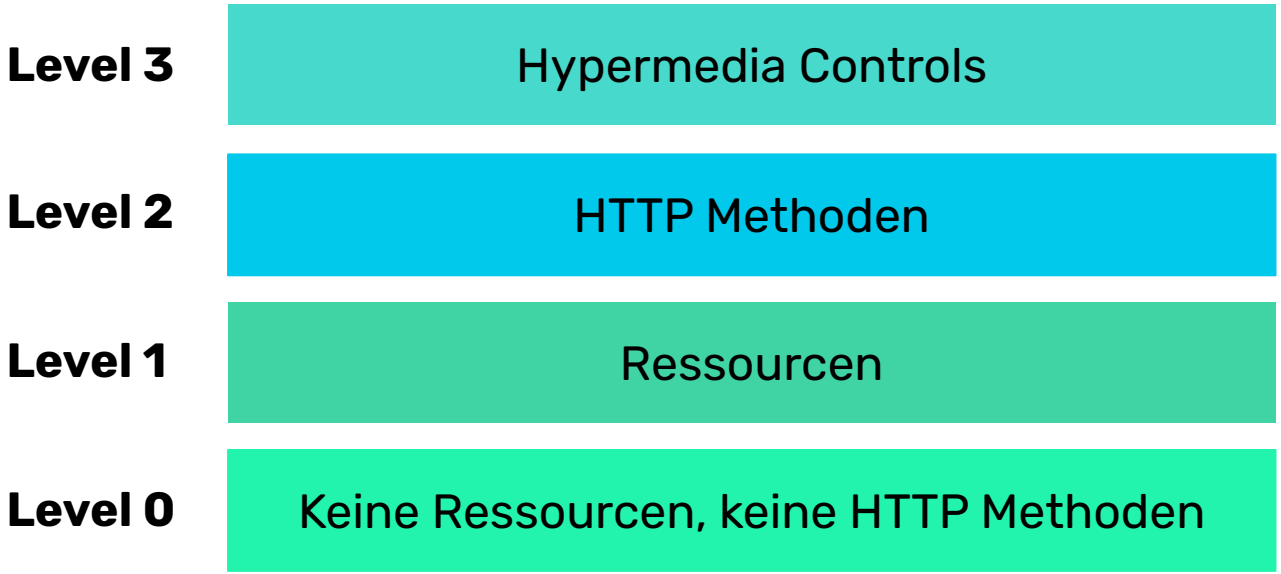
“Wir brauchen dringend eine API.”


API = REST

Die REST Prinzipien

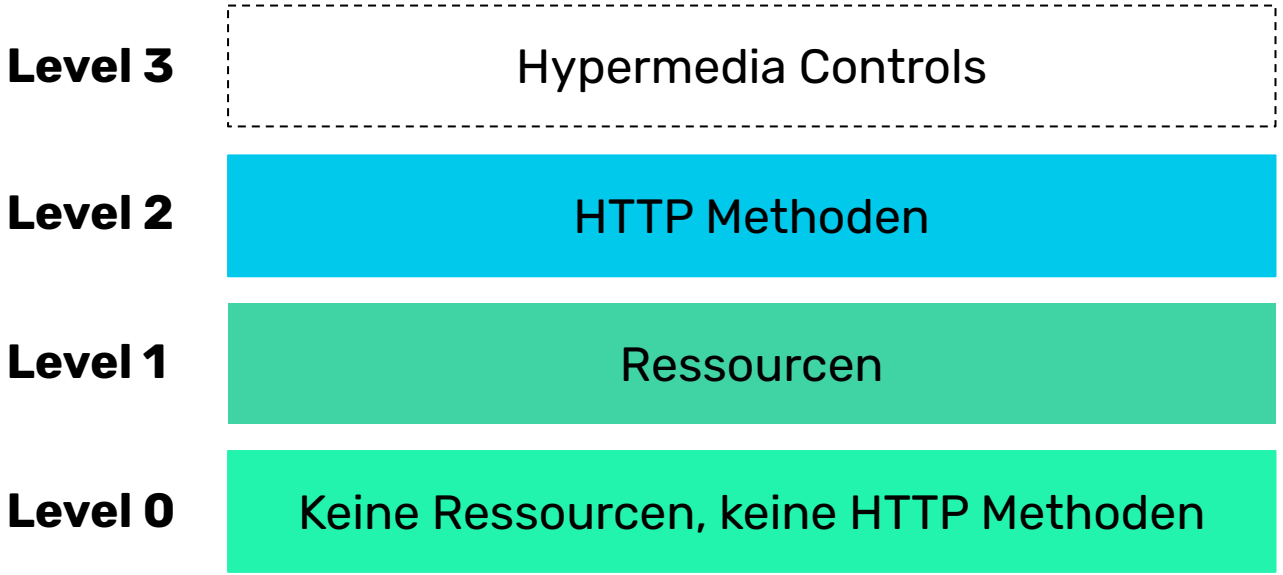
- Client-Server Architektur
- Mehrschichtiges System
- Zustandslosigkeit
- Caching
- Code-on-Demand (optional)
- **Einheitliche Schnittstelle**

Richardson Maturity Model



https://www.crummy.com/writing/speaking/2008-QCon/act3.html

Richardson Maturity Model



https://www.crummy.com/writing/speaking/2008-QCon/act3.html

Wir machen kein REST,
wir machen HTTP-basierte
JSON Schnittstellen.





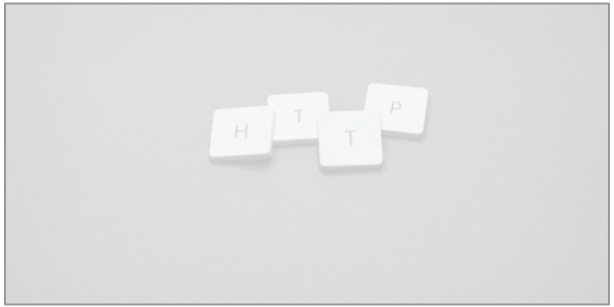

H T T P



RULES







H T T P



RULES


REST APIs sind das Allheilmittel!


TIMEOUT

Welches Ziel wird insgesamt verfolgt?

Wer agiert um das Ziel zu erreichen?

Gibt es Vor- oder Nach-bedingungen?

Resource / Hypermedia

Tunnel

Query

Event-Basiert

Welche Größe haben die Daten?

Werden die Daten gespeichert?

Müssen die Daten aktuell sein?

Lessons learned

Use Case identifizieren

Warum wird die API überhaupt gebraucht? Was will die Person, die die API verwendet damit erreichen?

Integration Enabling

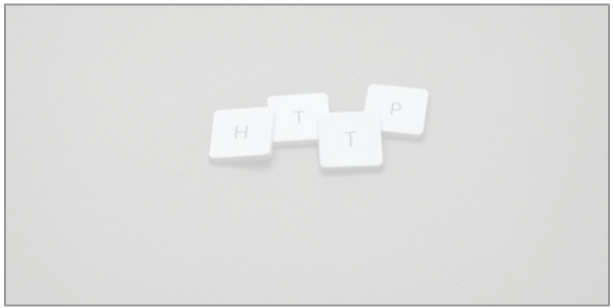
Alle Beteiligten kennen gängige Interaction Patterns und können sie zumindest grundlegend anwenden.

Fragen stellen

Die richtigen Fragen zu Vorkenntnissen, Daten-verwendung und Anwendung sind entscheidend.






H T T P



RULES


Was ist eine Ressource?

Ich habe schon ein Datenmodell!

Request Body


```
"contractId": "12345",
"contractType": "permanent",
"startDate": "2022-01-01",
"endDate": "2022-12-31",
"employeeName": "John Doe",
"employeeId": "123456789",
"department": "IT",
"jobTitle": "Software Engineer",
"salary": 50000,
"bonus": 2000,
"benefits": [
  {
    "type": "health insurance",
    "value": 1000
  },
  {
    "type": "life insurance",
    "value": 5000
  }
],
"terminationReason": "resignation",
"terminationDate": "2023-01-01",
"severancePay": 10000,
"terminationNoticePeriod": 60,
"furloughDays": 14,
"furloughRate": 0.5,
"vacationHours": 20,
"sickLeaveHours": 10,
"juryDuty": true,
"overtimeHours": 80,
"holidayPay": 1000,
"holidayAccrualRate": 0.5,
"bonusPayoutMonth": "January",
"bonusPayoutDate": "2023-01-01",
"bonusPayoutAmount": 10000,
"bonusPayoutFrequency": "monthly",
"performanceReview": true,
"performanceReviewComments": "Excellent job! Keep it up.",
"performanceReviewRating": 5,
"promotionDate": "2023-01-01",
"promotionTo": "Senior Software Engineer",
"promotionSalaryIncrease": 2000,
"promotionBenefits": [
  {
    "type": "health insurance",
    "value": 5000
  },
  {
    "type": "life insurance",
    "value": 10000
  }
],
"relocationExpenses": 2000,
"relocationDestination": "New York City",
"relocationReason": "promotion",
"relocationStartDate": "2023-01-01",
"trainingProgram": true,
"trainingProgramName": "Agile Development Practices",
"trainingProgramDuration": 6,
"trainingProgramProvider": "Scrum.org",
"trainingProgramLocation": "Online",
"trainingProgramCertificate": true,
"trainingProgramCertificateIssuer": "Scrum.org",
"trainingProgramCertificateDate": "2023-01-01",
"complianceTraining": true,
"complianceTrainingProvider": "OSHA",
"complianceTrainingLocation": "Online",
"complianceTrainingCompletionDate": "2023-01-01",
"confidentialityAgreement": true,
"confidentialityAgreementName": "Confidentiality Agreement",
"confidentialityAgreementDate": "2023-01-01",
"nonCompeteClause": true,
"nonCompeteClauseStartDate": "2023-01-01",
"nonCompeteClauseEndDate": "2025-01-01",
"nonCompeteClauseEmployerName": "XYZ Corporation",
"nonCompeteClauseJobTitle": "Software Engineer",
"restrictedShares": true,
"restrictedSharesNumber": 5,
"restrictedSharesValue": 10000,
"restrictedSharesVestedDate": "2023-01-01",
"restrictedSharesUnvestedDate": "2024-01-01",
"restrictedSharesHoldingPeriod": 5,
"restrictedSharesUnvestedAmount": 5000,
"nonDisclosureAgreement": true,
"nonDisclosureAgreementName": "Non Disclosure Agreement",
"nonDisclosureAgreementDate": "2023-01-01",
"employmentContract": true,
"employmentContractType": "permanent",
"employmentContractStartDate": "2022-01-01",
"employmentContractEndDate": "2024-01-01",
"employmentContractTerminationReason": "resignation",
"employmentContractTerminationDate": "2023-01-01",
"employmentContractSeverancePay": 10000,
"employmentContractNoticePeriod": 60,
"competitiveRewards": true,
"competitiveRewardsType": "performance bonus",
"competitiveRewardsAmount": 5000,
"competitiveRewardsFrequency": "monthly",
"performanceGoals": true,
"performanceGoalsType": "specific results",
"performanceGoalsMetrics": [
  {
    "type": "meeting attendance",
    "value": 80
  },
  {
    "type": "project completion",
    "value": 95
  }
],
"performanceGoalsTargets": [
  {
    "type": "meeting attendance",
    "target": 100
  },
  {
    "type": "project completion",
    "target": 100
  }
],
"performanceGoalsComments": "Excellent job! Keep it up.",
"performanceGoalsRating": 5,
"teamManagement": true,
"teamManagementType": "team player",
"teamManagementComments": "Excellent team player!",
"teamManagementRating": 5,
"jobDescriptions": [
  {
    "type": "software engineering",
    "value": "develop and maintain software systems"
  },
  {
    "type": "system administration",
    "value": "manage and configure computer systems"
  }
],
"jobResponsibilities": [
  {
    "type": "software engineering",
    "value": "develop and maintain software systems"
  },
  {
    "type": "system administration",
    "value": "manage and configure computer systems"
  }
],
"jobBenefits": [
  {
    "type": "health insurance",
    "value": 1000
  },
  {
    "type": "life insurance",
    "value": 5000
  }
],
"jobSkills": [
  {
    "type": "programming languages",
    "value": ["Python", "Java"]
  },
  {
    "type": "operating systems",
    "value": ["Windows", "Linux"]
  }
],
"jobQualifications": [
  {
    "type": "education",
    "value": "Bachelor's degree in computer science"
  },
  {
    "type": "experience",
    "value": "5+ years of experience as a software engineer"
  }
],
"jobCertifications": [
  {
    "type": "certification",
    "value": "C++ certification"
  },
  {
    "type": "certification",
    "value": "Java certification"
  }
],
"jobTraining": [
  {
    "type": "training",
    "value": "Agile methodologies training"
  },
  {
    "type": "training",
    "value": "DevOps training"
  }
]
}
```


Brauchen wir dieses Feld?
Brauchen wir verschachtelte Daten?
Können wir dieses Feld automatisch füllen?

Lessons learned

Intern != Extern

Das interne Datenmodell des Backend-Systems und das externe Datenmodell müssen nicht gleich sein.

Komplexität reduzieren

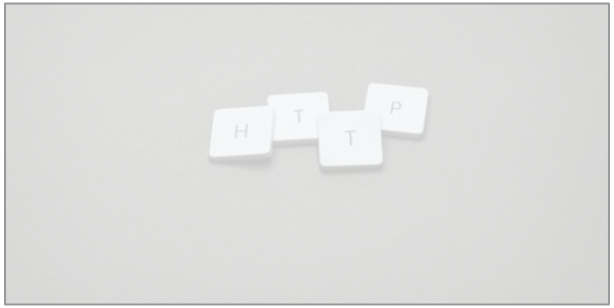
Die Komplexität des Datenmodells muss für bestimmte Kontexte drastisch reduziert werden.

Fachlichkeit

Die Anforderungen können nicht einfach über den Zaun geworfen werden, Entwicklungsteams brauchen Zugang zu Expert*innen.









RULES


Die Nutzer*innen verstehen das!

Je generischer desto besser!


DU BIST
NICHT
DER USER

EMPATHIC CONVERSATION

JOB DOMAIN-DRIVEN DESIGN

SHADOWING DOMAIN

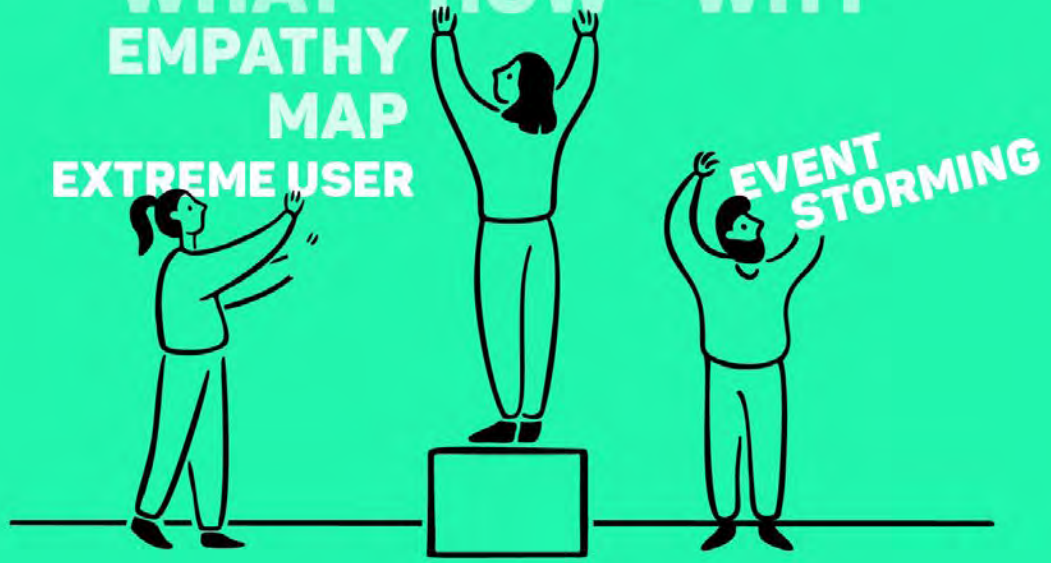
PERSONAS STORYTELLING

WHAT – HOW – WHY

EMPATHY MAP

EXTREME USER

EVENT STORMING




MEETING

Lessons learned

Du bist nicht der User

Selbstreferentielles Design muss vermieden werden, auch wenn wir als Entwickler*innen glauben zu wissen was die Nutzer*innen brauchen.

Zielgruppen festlegen

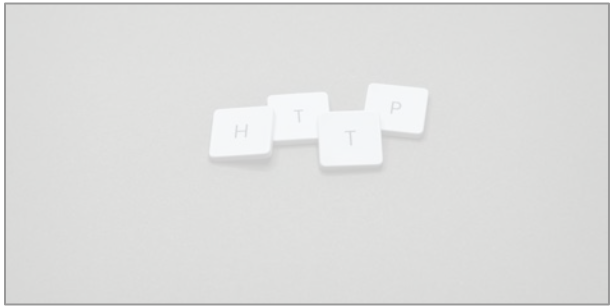
Wer sitzt auf der anderen Seite und verwendet die API? Was braucht diese Person um die API bestmöglich einsetzen zu können?

Austausch fördern

Unterschiedliche Teams zusammenzubringen kann das Verständnis füreinander erhöhen.







H T T P



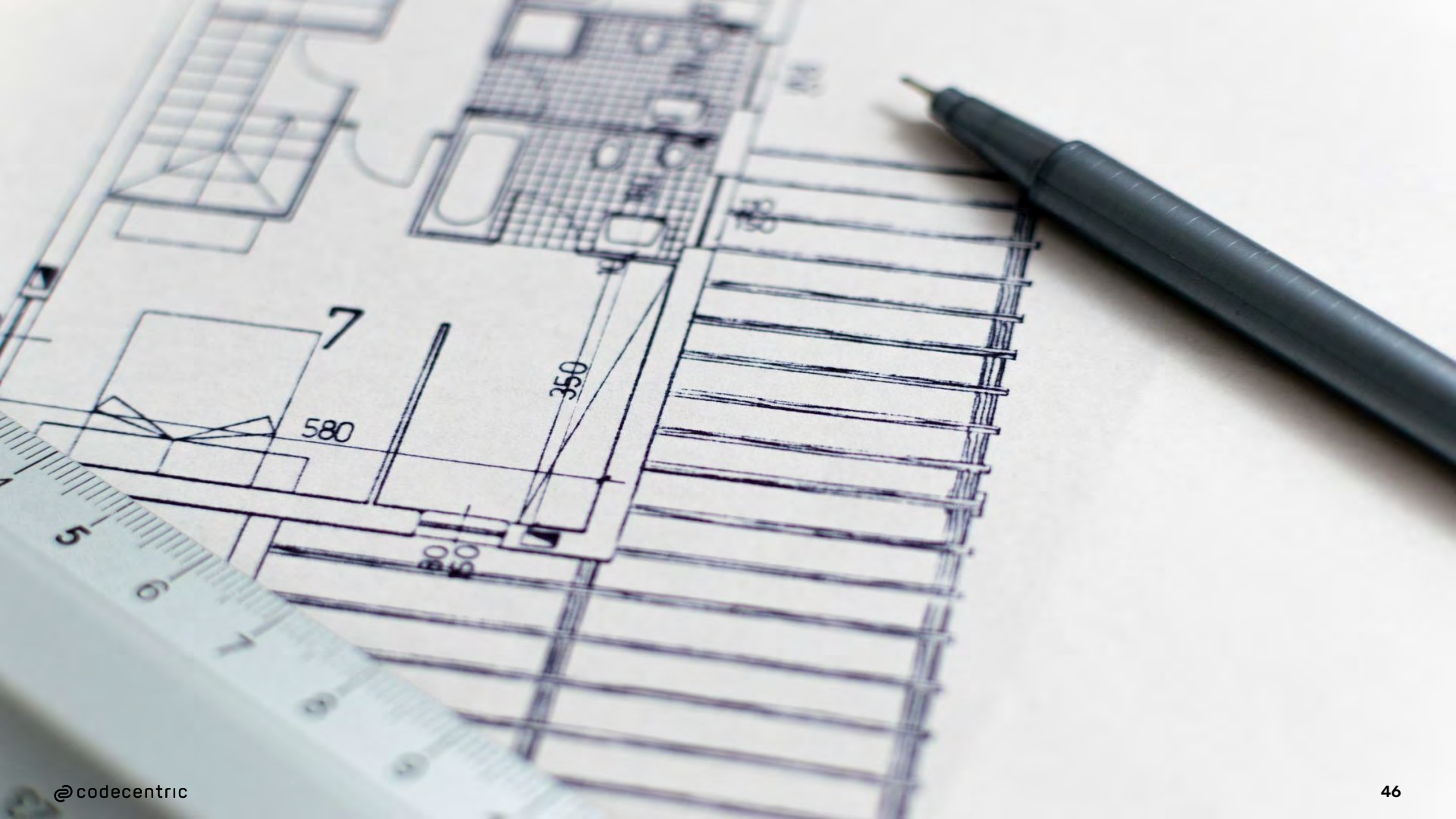
RULES


Welche Dokumentation?

Wer braucht schon ein Design?

Response Body


```
{
    "ID": "12345",
    "CTD_D": "/Date(1354665600000)/",
    "CTD_T": "PT14H01M00S",
    "LGR_No_ST": null,
    "AC_Flag": "",
    "BL_Flag": "X",
    "LGR_St_Txt": []
}
```


7
580
350


id string
created_at string<date-time>
location_number integer or null
activated boolean
blocked boolean
long_texts array[object]
value_de string
value_en string
key string

Response Body


```
{
    "id": "12345",
    "created_at": "1970-01-01T14:01:00",
    "location_number": null,
    "activated": false,
    "blocked": true,
    "long_texts": []
}
```

ISO-Normen und RFCs

- Datum & Zeit: RFC 3339 & ISO 8601
- Sprache: RFC 3066 & ISO 639
- Währung: ISO 4217

Lessons learned

API Design first

Gerade wenn die Erfahrung mit APIs in Entwicklungsteams gering ist, lohnt sich der API Design first Ansatz um das Design iterativ zu entwickeln.

Datentypen nutzen

Datentypen müssen richtig verwendet werden und entsprechend auch in der API Definition definiert werden.

An Standards halten

Datumsangaben, Zeitangaben, Währungsangaben, usw. sollten sich an gängigen RFCs orientieren.






H
T
T
P



RULES


Was ist nochmal HTTP?

GET /users Get list of users

Retrieve a list of users. Use the request body to filter the list of users by all fields of the user.

Parameters Try it out

No parameters

Request body application/json

Example Value | Schema

```
{
  "id": 0,
  "firstName": "string",
  "lastName": "string",
  "email": "user@example.com",
  "dateOfBirth": "1997-10-31",
  "emailVerified": true,
  "createDate": "2024-05-12"
}
```

Was sagt die HTTP Spezifikation dazu?

A message-body MUST NOT be included in a request if the specification of the request method [...] does not allow sending an entity-body in requests. (4.3)

The GET method means retrieve whatever information [...] is identified by the Request-URI. (9.3)

➔ Nicht explizit verboten, aber auch nicht explizit erlaubt!

siehe https://www.rfc-editor.org/rfc/rfc2616

Resource / Hypermedia

Tunnel

Query

Event-Basiert


DEAD
END

Lessons learned

Standards kennen

Wenn ich einen Standard nutze, muss ich ihn ausreichend kennen und verstehen.

Pattern hinterfragen

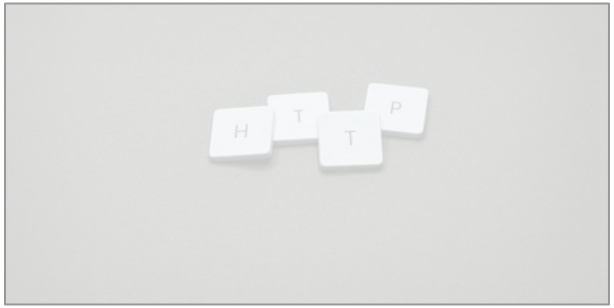
Wenn der Standard nicht passt, habe ich eventuell die falsche Technologie oder sogar das falsche Interaction Pattern ausgewählt.

Technologie kennen

Was kann die Technologie die ich einsetze wirklich? Kann sie vom Standard abweichen wenn ich das brauche?






H
T
T
P



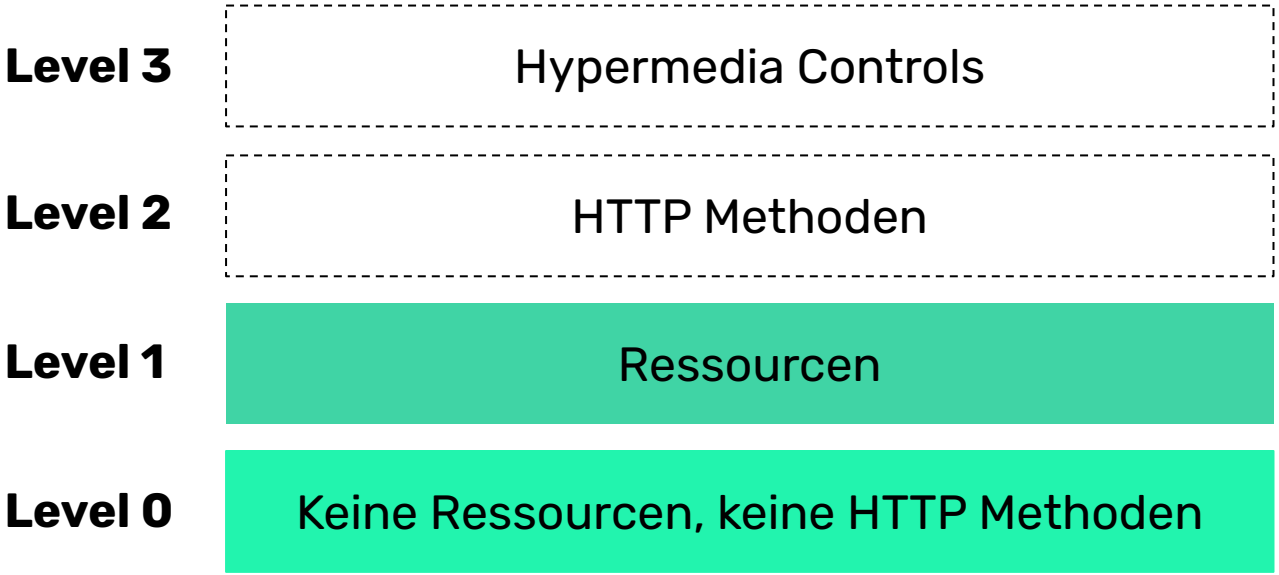
RULES


POST funktioniert immer!

Contracts

POST	`/contracts/create`	Create new contract
POST	`/contracts/change`	Change header fields of contract
POST	`/contracts/update-status`	Update status of contract
POST	`/contracts/delete`	Mark contract as deleted
POST	`/contracts/postpone`	Postpone contract
POST	`/contracts/extend`	Extend contract length

Richardson Maturity Model



https://www.crummy.com/writing/speaking/2008-QCon/act3.html


FAHRSCHULE

Lessons learned

Redesign

Wenn die API noch nicht in Produktion ist, kann ein Redesign hilfreich sein.

Schulungen

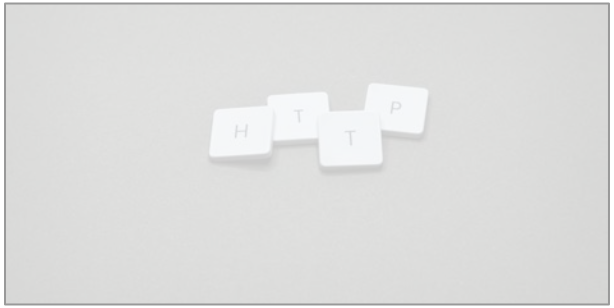
Wenn die Standards nicht bekannt sind, hilft eine Grundlagenschulung für alle Entwicklungsteams.

Enabling Team

Ein API Enabling Team unterstützt Entwicklungsteams beim Design ihrer ersten APIs.






H
T
T
P



RULES


Jeder Aufruf ist erfolgreich!

Responses

Code	Description	Links
200	OK	*No links*


RULES

RFC 9457: application/problem+json

Repräsentation für Fehler mit folgenden Feldern:

- type (string)
- status (number)
- title (string)
- detail (string)
- instance (string)

```
{
  "type": "https://example.com/probs/out-of-credit",
  "title": "You do not have enough credit.",
  "detail": "Your current balance is 30, but that costs 50.",
  "instance": "/account/12345/msgs/abc",
  "balance": 30,
  "accounts": ["/account/12345", "/account/67890"]
}
```

siehe https://www.rfc-editor.org/rfc/rfc9457.html


BIKE SHARE
TORONTO

Lessons learned

API Guidelines

Die API Guidelines definieren welche Statuscodes wann zurückgegeben werden müssen.

An Standards halten

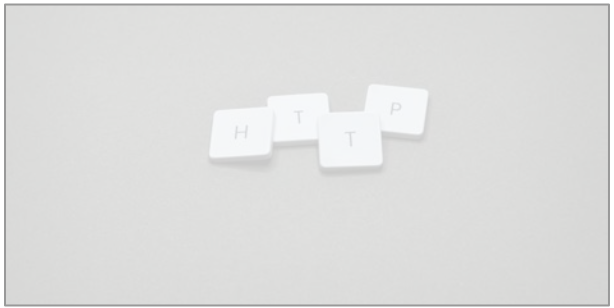
Auch für die Repräsentation von Fehlern gibt es Standardformate, die für einheitliche Schnittstellen Voraussetzung sind.

Geteilte Funktionalität

Fehlerbehandlung kann über globalen Funktionen oder/und im API Management vereinheitlicht werden.






H T T P



RULES


API Guidelines lese ich nicht!

API Guidelines

Table of Contents

Zalando RESTful API and Event Guidelines

GitHub Repository as part of Zalando SE Opensource

Table of Contents

https://opensource.zalando.com/restful-api-guidelines/


QUALITY CONTROL
CONTROL
QUALITY
CONTROL

Lessons learned

Adaptieren

API Guidelines müssen individuell und passend sein. Sie ohne Adaption zu kopieren führt zu Frustration bei Entwicklungsteams.

Kurz & knapp

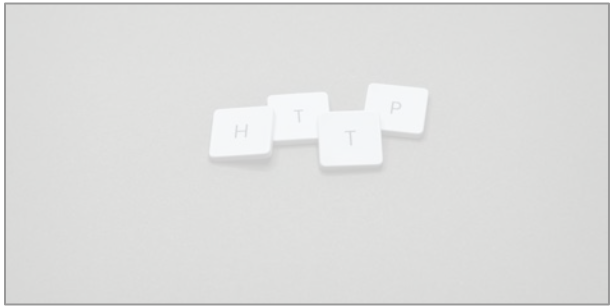
API Guidelines müssen kurz und knapp das wichtigste zusammenfassen. Niemand liest lange Guidelines!

Validierung

Eine automatisierte Validierung der API Guidelines unterstützt Entwicklungsteams dabei sie einzuhalten.






H T T P



RULES


Warum soll ich testen?

Lessons learned

Tests automatisieren

Durch Automatisierung haben die Teams keine “Ausrede” mehr nicht zu testen, da es wenig Zeit kostet und einfach ist.

Kundenverständnis

Die Entwicklungsteams dürfen ihre Nutzer*innen nicht als lästiges Übel begreifen, sondern müssen mit Kundenfokus arbeiten.

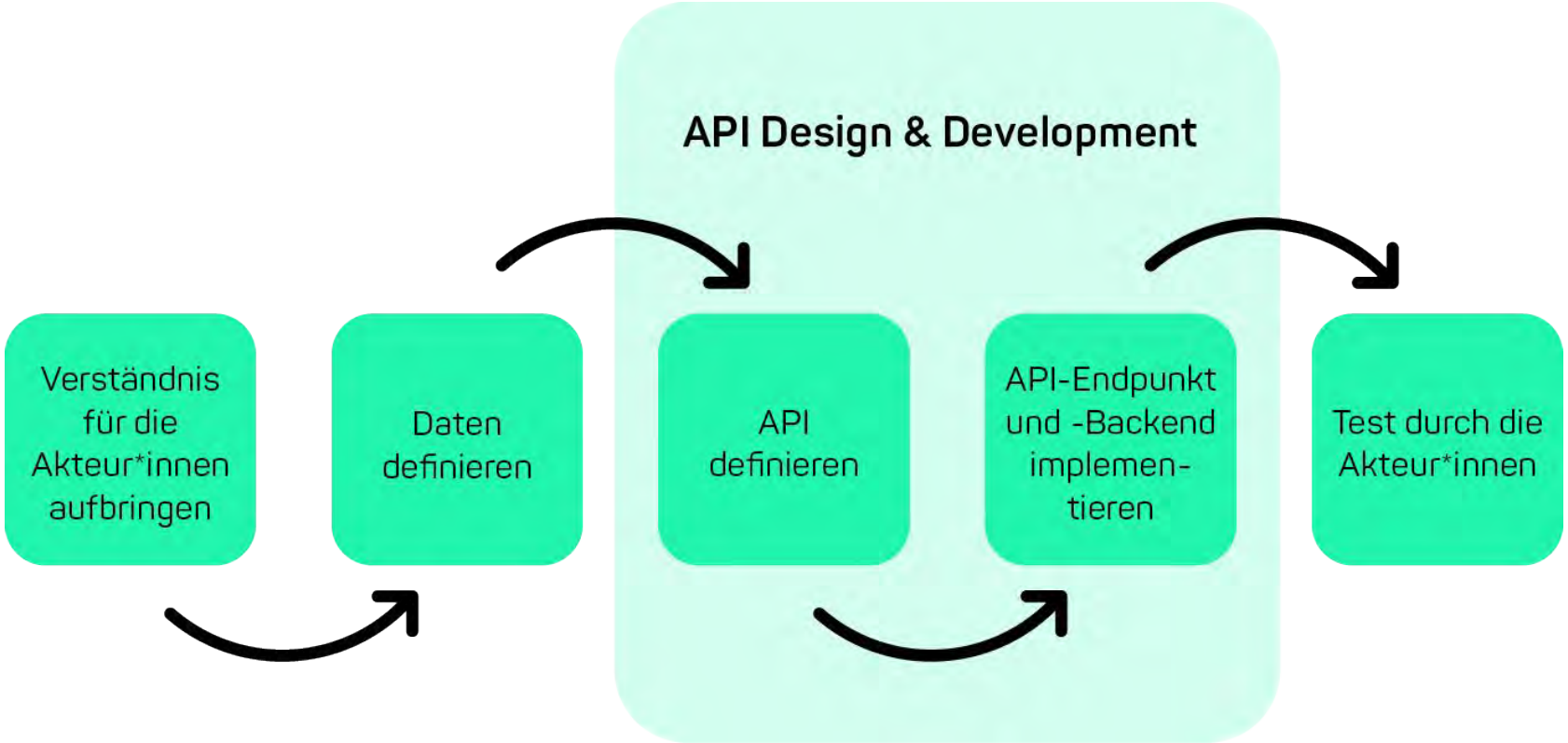
Erfolge feiern

Wenn die Nutzer*innen beim Testen dann zufrieden sind, muss dieser Erfolg und die Zeitersparnis dadurch hervorgehoben werden.


H
T
T
P


RULES

API Thinking



Misserfolge vermeiden durch

- Schulungen
- API Thinking
- Enabling Teams
- Automatisierung

An Standards halten und nur bewusst davon abweichen!

Miri (sie/ihr)

Dr. Miriam Greis
Lead API Consultant
miriam.greis@codecentric.de
https://www.linkedin.com/in/miriam-greis