2.9 Prozessautomatisierung mit n8n

n8n ist eine Plattform zur Automatisierung von Arbeitsabläufen.

Ein n8n-Workflow verbindet verschiedene Systeme und führt festgelegte Schritte automatisch aus.

Typische Einsatzbereiche:

- Daten zwischen SaaS-Systemen übertragen
- Benutzer-Onboarding vorbereiten
- API-Anfragen ausführen
- Dateien verarbeiten
- Benachrichtigungen senden
- Daten prüfen und umwandeln
- regelmäßige Berichte erstellen
- Fehler erkennen und melden

> n8n verbindet Trigger, Daten, Bedingungen und Aktionen zu einem automatisierten Workflow.

Lernziele

Nach dieser Seite solltest du erklären können:

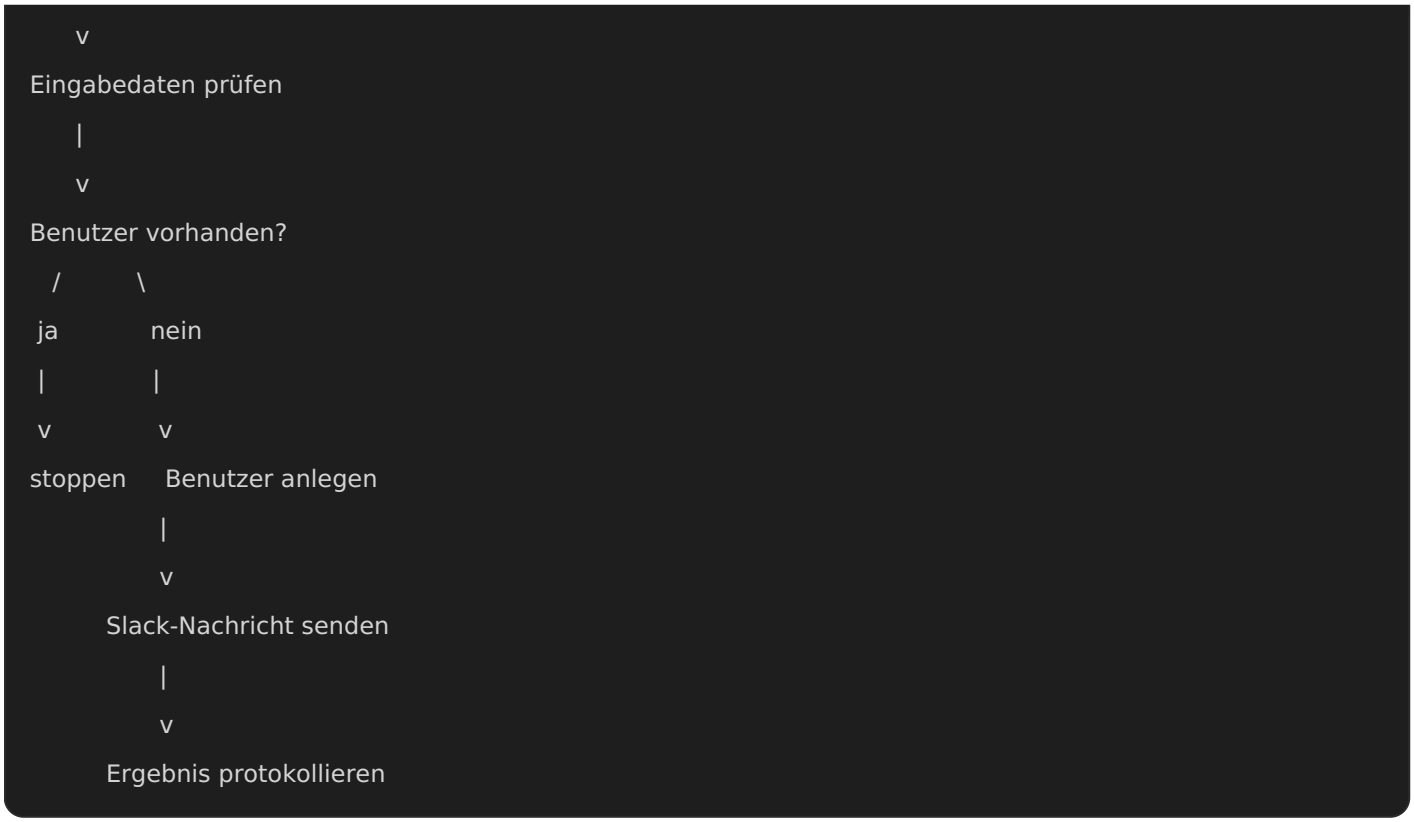
- wie ein n8n-Workflow aufgebaut ist
- was Nodes, Trigger und Verbindungen sind
- wie Daten zwischen Nodes übertragen werden
- wie Expressions verwendet werden
- wie Credentials geschützt gespeichert werden
- wie man Workflows testet und veröffentlicht
- wie Fehler behandelt und Ausführungen geprüft werden

Grundaufbau eines Workflows

Ein Workflow besteht aus mehreren miteinander verbundenen Nodes.

Beispiel:

```
Webhook Trigger
      |
```



Jede Node übernimmt eine bestimmte Aufgabe.

Node

Eine Node ist ein einzelner Verarbeitungsschritt.

Typische Nodes:

Node-Art	Aufgabe
Trigger Node	startet den Workflow
Action Node	führt eine Aktion aus
HTTP Request	ruft eine API auf
Edit Fields	erstellt oder verändert Felder
If	prüft eine Bedingung
Switch	unterscheidet mehrere Fälle
Code	verarbeitet Daten mit JavaScript
Webhook	empfängt HTTP-Anfragen
Schedule Trigger	startet nach Zeitplan

> Eine Node erhält Daten, verarbeitet sie und gibt ein Ergebnis an die nächste Node weiter.

Trigger Node

Jeder automatisierte Workflow benötigt einen Startpunkt.

Mögliche Trigger:

- Webhook
- Zeitplan
- neue E-Mail
- neuer CRM-Datensatz
- geänderte Datei
- manuelle Ausführung
- Ereignis aus einer anderen Anwendung

Beispiel:

```
Neuer Mitarbeiter wird freigegeben
            |
            v
      Workflow startet
```

Ohne Trigger wird ein Workflow nicht automatisch ausgeführt.

Action Node

Eine Action Node führt eine konkrete Aktion aus.

Beispiele:

- Benutzerkonto erstellen
- Nachricht an Slack senden
- Asana-Aufgabe anlegen
- Datenbankeintrag speichern
- Datei verschieben
- API-Daten abrufen
- E-Mail versenden

Ein Workflow kann mehrere Action Nodes nacheinander verwenden.

Verbindungen zwischen Nodes

Verbindungen bestimmen die Reihenfolge der Verarbeitung.

Beispiel:

```
Trigger
  |
  v
Daten prüfen
  |
  v
API aufrufen
  |
  v
Antwort auswerten
```

Die Ausgabe einer Node wird normalerweise zur Eingabe der nächsten Node.

Datenstruktur in n8n

n8n verarbeitet Daten meistens als JSON.

Beispiel:

```
{
  "name": "Max Mustermann",
  "department": "IT",
  "active": true
}
```

Die nächste Node kann auf diese Werte zugreifen.

Beispiel:

```
{{$json.name}}
```

Ergebnis:

```
Max Mustermann
```

Weiteres Beispiel:

```
{{$json.department}}
```

Ergebnis:

```
IT
```

Expressions

Expressions ermöglichen dynamische Werte.

Sie beginnen häufig mit:

```
{{ ... }}
```

Beispiel:

```
{{$json.email}}
```

Der Wert wird aus den Eingangsdaten übernommen.

Eine Nachricht könnte so aufgebaut werden:

```
Benutzer {{$json.name}} wurde angelegt.
```

Bei den Eingangsdaten:

```
{
  "name": "Max Mustermann"
}
```

entsteht:

```
Benutzer Max Mustermann wurde angelegt.
```

Datenmapping

Datenmapping ordnet Felder eines Quellsystems den Feldern eines Zielsystems zu.

Beispiel:

Quellsystem	Zielsystem
`firstName`	`givenName`
`lastName`	`familyName`
`mail`	`primaryEmail`
`department`	`organization.department`

Quellsystem:

```
{
 "firstName": "Max",
 "lastName": "Mustermann",
 "mail": "max.mustermann@example.com"
}
```

Zielsystem:

```
{
 "givenName": "Max",
 "familyName": "Mustermann",
 "primaryEmail": "max.mustermann@example.com"
}
```

> Beim Datenmapping werden Werte übernommen, obwohl die Systeme unterschiedliche Feldnamen verwenden.

Edit Fields Node

Mit der Edit Fields Node können Daten:

- ausgewählt
- umbenannt
- ergänzt
- entfernt
- neu aufgebaut

werden.

Beispiel:

Eingangsdaten:

```
{
  "firstName": "Max",
  "lastName": "Mustermann",
  "department": "IT",
  "internalNote": "nicht übertragen"
}
```

Ausgangsdaten:

```
{
  "fullName": "Max Mustermann",
  "department": "IT"
}
```

Damit werden nur die benötigten Daten an das Zielsystem übertragen.

If Node

Eine If Node prüft eine Bedingung.

Beispiel:

> Ist der Benutzer aktiv?

```
active = true?
   /      \
 ja        nein
 |          |
 v          v
weiter    Workflow beenden
```

Mögliche Prüfungen:

- Wert ist gleich
- Wert ist ungleich
- Text enthält Zeichenfolge
- Zahl ist größer oder kleiner
- Feld ist vorhanden
- Datum liegt vor oder nach einem Zeitpunkt
- Boolean ist `true` oder `false`

Switch Node

Eine Switch Node eignet sich für mehrere mögliche Fälle.

Beispiel:

```
Abteilung prüfen
  /     |      \
 v      v       v
IT    Vertrieb  Buchhaltung
 |      |       |
 v      v       v
IT-Gruppe CRM     Finanzsystem
```

Die Switch Node ist übersichtlicher als viele verschachtelte If-Nodes.

HTTP Request Node

Die HTTP Request Node kann APIs aufrufen.

Benötigte Angaben:

- HTTP-Methode
- Endpoint
- Authentifizierung
- Header
- Query-Parameter
- Request-Body
- erwartetes Antwortformat

Beispiel:

```
POST https://api.example.com/users
```

```
Content-Type: application/json

{
  "name": "Max Mustermann",
  "department": "IT"
}
```

Die Response kann anschließend von weiteren Nodes verarbeitet werden.

Credentials

Credentials enthalten Zugangsdaten für externe Systeme.

Beispiele:

- API-Key
- Bearer-Token
- OAuth-Verbindung
- Benutzername und Passwort
- Client ID und Client Secret

Credentials sollten in der n8n-Credential-Verwaltung gespeichert werden.

Sie sollten nicht direkt stehen in:

- Workflow-Namen
- Beschreibungen
- Code-Nodes
- öffentlichen Dateien
- Screenshots
- Wiki-Seiten
- Git-Repositories

> Workflow-Logik und geheime Zugangsdaten sollten voneinander getrennt werden.

Manuelle Ausführung

Ein Workflow kann manuell gestartet werden.

Das ist besonders geeignet für:

- Aufbau eines neuen Workflows
- Tests
- Fehlersuche
- Prüfung von Daten
- einmalige administrative Aufgaben

Beim Testen sollte möglichst mit ungefährlichen Testdaten gearbeitet werden.

Veröffentlichter Workflow

Damit ein Workflow automatisch auf Trigger reagieren kann, muss er entsprechend aktiviert beziehungsweise veröffentlicht sein.

Vorher sollten geprüft werden:

- funktionieren alle Nodes?
- sind die Credentials korrekt?
- stimmen die Endpoints?
- werden Produktivdaten verändert?
- existiert eine Fehlerbehandlung?
- können doppelte Datensätze entstehen?
- ist die Protokollierung ausreichend?

> Ein getesteter Workflow ist nicht automatisch ein sicherer Produktivworkflow.

Test- und Produktivdaten

Testdaten sollten klar von Produktivdaten getrennt werden.

Mögliche Maßnahmen:

- Testkonten verwenden
- eigene Testprojekte anlegen
- Test-Endpoints verwenden
- Benachrichtigungen an Testkanäle senden
- Löschaktionen deaktivieren
- Änderungen zuerst manuell freigeben

Beispiel:

```
Testworkflow
    |
```

```
    v
Slack-Kanal #it-test
```

Statt:

```
Produktivworkflow
    |
    v
Slack-Kanal #allgemein
```

Workflow-Ausführung

Jeder Workflow-Lauf wird als Execution bezeichnet.

Eine Ausführung kann sein:

- erfolgreich
- fehlgeschlagen
- manuell gestartet
- automatisch gestartet
- noch in Bearbeitung
- abgebrochen

Die Ausführungsdaten helfen bei der Fehlersuche.

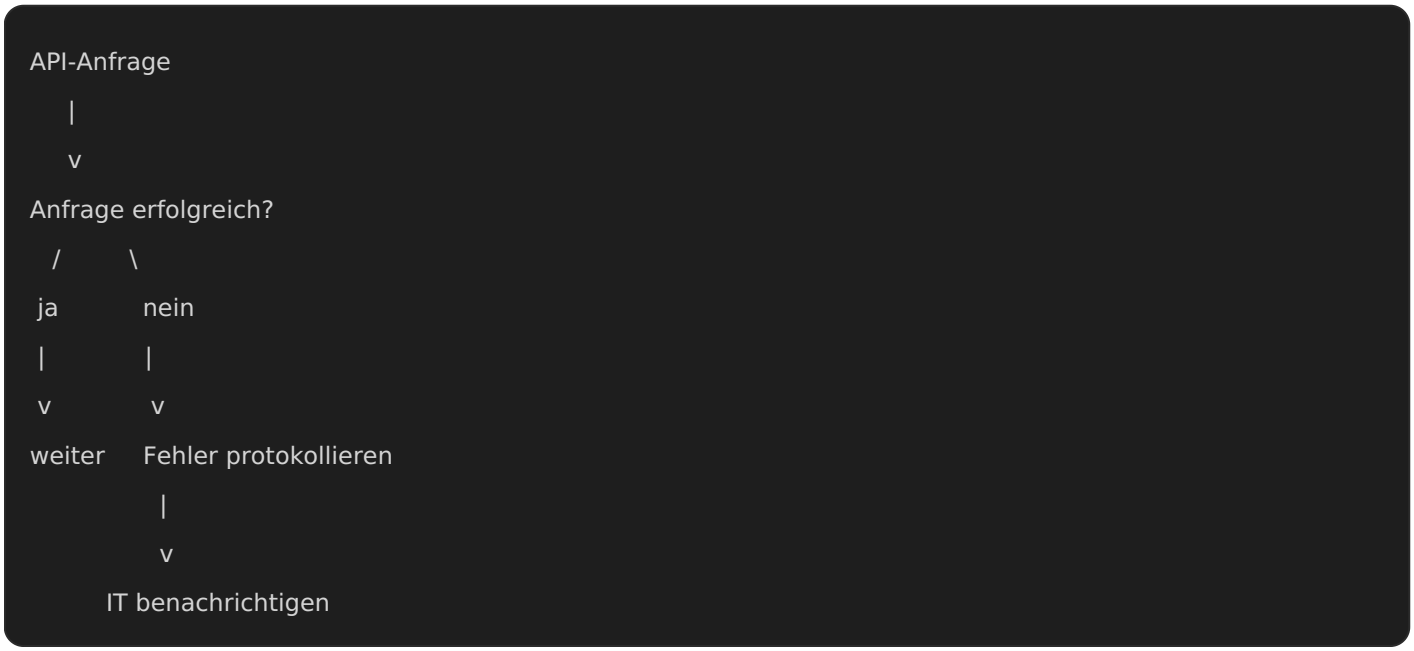
Geprüft werden können:

- verwendete Eingangsdaten
- Ausgabe jeder Node
- fehlgeschlagener Schritt
- Fehlermeldung
- Dauer
- Startzeit
- Status

Fehlerbehandlung

Ein professioneller Workflow benötigt einen Fehlerpfad.

Beispiel:



Mögliche Fehler:

- API nicht erreichbar
- Token abgelaufen
- Berechtigung fehlt
- JSON ungültig
- Pflichtfeld fehlt
- Rate Limit erreicht
- Zielsystem antwortet zu langsam
- Datensatz existiert bereits

Error Workflow

Ein separater Error Workflow kann auf fehlgeschlagene Ausführungen reagieren.

Mögliche Aktionen:

- Fehlermeldung an Slack senden
- E-Mail an die IT senden
- Ticket erstellen
- Fehlerdaten speichern
- zuständige Person informieren

Beispielmeldung:

```
Workflow: Mitarbeiter-Onboarding
Status: fehlgeschlagen
Node: Benutzer anlegen
```

```
Fehler: 403 Forbidden
Zeitpunkt: 18.07.2026 10:15 Uhr
```

Geheime Zugangsdaten dürfen dabei nicht mitgesendet werden.

Stop And Error

Ein Workflow sollte gezielt abgebrochen werden, wenn wichtige Bedingungen nicht erfüllt sind.

Beispiele:

- E-Mail-Adresse fehlt
- Mitarbeiter-ID ist ungültig
- Abteilung ist unbekannt
- Benutzer existiert bereits
- notwendige Freigabe fehlt

Beispiel:

```
Pflichtfelder prüfen
      |
      v
E-Mail vorhanden?
  /       \
 ja         nein
 |          |
 v          v
weiter    Stop And Error
```

Dadurch wird verhindert, dass fehlerhafte Daten weiterverarbeitet werden.

Retry

Vorübergehende Fehler können einen erneuten Versuch erlauben.

Geeignet für Retry:

- `429 Too Many Requests`
- `502 Bad Gateway`
- `503 Service Unavailable`
- `504 Gateway Timeout`
- kurzfristiger Netzwerkfehler

Nicht unverändert wiederholen:

- `400 Bad Request`
- `401 Unauthorized`
- `403 Forbidden`
- ungültige Pflichtfelder

Ein Retry sollte:

- begrenzt sein
- eine Wartezeit verwenden
- protokolliert werden
- keine doppelten Datensätze erzeugen

Doppelte Ausführungen verhindern

Ein Workflow kann versehentlich mehrfach gestartet werden.

Mögliche Schutzmaßnahmen:

- eindeutige Mitarbeiter-ID prüfen
- E-Mail-Adresse vor dem Anlegen suchen
- Event-ID speichern
- Idempotency-Key verwenden
- bereits verarbeitete Vorgänge markieren

Beispiel:

```
Benutzer suchen
      |
      v
bereits vorhanden?
  /       \
 ja         nein
 |          |
 v          v
stoppen    anlegen
```

Teilweise erfolgreiche Workflows

Ein Workflow kann teilweise erfolgreich sein.

Beispiel:

Schritt	Ergebnis
Google-Konto anlegen	erfolgreich
Gruppe zuweisen	erfolgreich
Slack-Zugang anlegen	fehlgeschlagen
IT-Aufgabe erstellen	nicht ausgeführt

Der Workflow darf dann nicht melden:

> Alles erfolgreich.

Stattdessen sollte er melden:

> Das Benutzerkonto wurde angelegt. Die Slack-Einrichtung ist fehlgeschlagen und muss geprüft werden.

Praxisbeispiel: Mitarbeiter-Onboarding

Eingangsdaten:

```
{
  "employeeId": "EMP-105",
  "name": "Max Mustermann",
  "email": "max.mustermann@example.com",
  "department": "IT",
  "startDate": "2026-08-03"
}
```

Möglicher Workflow:

1. Webhook empfängt die Daten.
2. Pflichtfelder werden geprüft.
3. Mitarbeiter-ID wird auf doppelte Verarbeitung geprüft.
4. Benutzer wird über eine API gesucht.
5. Ist kein Benutzer vorhanden, wird ein Konto erstellt.
6. Passende Gruppen werden anhand der Abteilung gewählt.
7. Eine Aufgabe für die Hardware wird erstellt.
8. Die IT erhält eine Slack-Nachricht.
9. Das Ergebnis wird protokolliert.

10. Bei einem Fehler wird der Error Workflow gestartet.

Dokumentation

Ein Workflow sollte dokumentiert werden.

Wichtige Angaben:

- Name und Zweck
- verantwortliche Person
- verwendeter Trigger
- beteiligte Systeme
- benötigte Credentials
- Eingabedaten
- Verarbeitungsschritte
- Ausgabedaten
- Fehlerbehandlung
- Zeitplan
- Abhängigkeiten
- Testverfahren
- Wiederherstellungsverfahren

Ein verständlicher Workflow-Name wäre:

```
Mitarbeiter-Onboarding – Google Workspace und Slack
```

Weniger verständlich wäre:

```
Workflow 17 Kopie Neu Final 2
```

Gute Workflow-Regeln

- eindeutige Node-Namen verwenden
- nur benötigte Daten übertragen
- Credentials sicher speichern
- Eingabedaten validieren
- Fehlerpfade einbauen
- doppelte Ausführungen verhindern
- kritische Aktionen absichern
- Ergebnisse protokollieren
- Workflows vor Änderungen sichern
- Änderungen zuerst testen
- verantwortliche Person dokumentieren

Systematische Fehlersuche

Wenn ein n8n-Workflow fehlschlägt:

1. Wurde der richtige Trigger ausgelöst?
2. Ist der Workflow veröffentlicht beziehungsweise aktiv?
3. Welche Eingangsdaten wurden empfangen?
4. Sind alle Pflichtfelder vorhanden?
5. In welcher Node trat der Fehler auf?
6. Welche Fehlermeldung wird angezeigt?
7. Sind die Credentials gültig?
8. Ist der API-Endpoint erreichbar?
9. Welcher HTTP-Statuscode wurde zurückgegeben?
10. Entspricht die JSON-Struktur der API-Dokumentation?
11. Sind Expressions und Feldpfade korrekt?
12. Wurde der Workflow möglicherweise doppelt ausgeführt?
13. Kann der fehlgeschlagene Schritt sicher wiederholt werden?
14. Muss ein teilweise erfolgreicher Vorgang manuell korrigiert werden?

Wichtige Begriffe

Begriff	Bedeutung
Workflow	automatisierter Ablauf aus mehreren Schritten
Node	einzelner Verarbeitungsschritt
Trigger	startet den Workflow
Action	führt eine konkrete Aktion aus
Expression	verwendet dynamische Werte
Datenmapping	ordnet Felder verschiedener Systeme zu
Credential	sicher gespeicherte Zugangsdaten
Execution	einzelne Workflow-Ausführung
Error Workflow	verarbeitet fehlgeschlagene Ausführungen
Retry	erneuter Ausführungsversuch
Testdaten	ungefährliche Daten für die Prüfung
Produktivdaten	echte Unternehmensdaten

Gesamtmerksatz

> Ein n8n-Workflow beginnt mit einem Trigger und verarbeitet Daten über verbundene Nodes. Expressions und Datenmapping übertragen Werte zwischen den Schritten. Credentials schützen API-Zugänge. Professionelle Workflows benötigen Validierung, Fehlerbehandlung, Protokollierung, sichere Tests und Schutz vor doppelten Ausführungen.

Kontrollfragen

Was ist eine Node?

Ein einzelner Verarbeitungsschritt innerhalb eines Workflows.

Was ist eine Trigger Node?

Eine Node, die einen Workflow durch ein Ereignis oder einen Zeitplan startet.

Was ist eine Expression?

Ein dynamischer Ausdruck, der Daten aus vorherigen Nodes verwendet.

Was bedeutet Datenmapping?

Felder eines Quellsystems werden passenden Feldern eines Zielsystems zugeordnet.

Was sind Credentials?

Geschützt gespeicherte Zugangsdaten für externe Systeme.

Was ist eine Execution?

Eine einzelne Ausführung eines Workflows.

Warum sollte ein Error Workflow verwendet werden?

Damit Fehler automatisch protokolliert und zuständige Personen informiert werden.

Warum müssen doppelte Ausführungen verhindert werden?

Damit beispielsweise Benutzer, Aufgaben oder Nachrichten nicht mehrfach erstellt werden.

Warum sollten Test- und Produktivdaten getrennt werden?

Damit Tests keine unbeabsichtigten Änderungen an echten Unternehmensdaten verursachen.

Was sollte bei einer teilweise erfolgreichen Ausführung passieren?

Erfolgreiche und fehlgeschlagene Schritte müssen getrennt dokumentiert und gegebenenfalls manuell korrigiert werden.

Quellen

- n8n-Dokumentation – Workflows und Workflow-Komponenten
- n8n-Dokumentation – Nodes
- n8n-Dokumentation – Expressions
- n8n-Dokumentation – Credentials
- n8n-Dokumentation – Executions
- n8n-Dokumentation – Error Handling