

# Arbeitszeitbetrug

## PRAXISBERICHT

für das Modul

T4\_2000

des Studienganges Informatik / Angewandte Informatik

an der

Dualen Hochschule Baden-Württemberg Karlsruhe

von

**Rudi Regentonne**

Abgabedatum 32. Nozember 13 v. Chr.

Bearbeitungszeitraum	560 Stunden
Matrikelnummer	123456789
Kurs	TINF2XXBX
Ausbildungsfirma	Saftladen GmbH
	Bielefeld
Betreuer der Ausbildungsfirma	Andi Mauer
Gutachter der Studienakademie	Prof. Dr. Jürgen R.

## **Ehrenwörtliche Erklärung**

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe und diese Arbeit bei keiner anderen Prüfung mit gleichem oder vergleichbarem Inhalt vorgelegt habe und diese bislang nicht veröffentlicht wurde. Des Weiteren versichere ich, dass die eingereichte elektronische Fassung mit der gedruckten Ausfertigung übereinstimmt.

---

Bielefeld

30.01.2026

gez. Rudi Regentonne

---

Ort

Datum

Unterschrift

# **Zusammenfassung**

# Inhaltsverzeichnis

Zusammenfassung	III
Abkürzungsverzeichnis	VI
Tabellenverzeichnis	VII
Quellcodeverzeichnis	VII
Formelverzeichnis	VIII
1 Einleitung	9
2 Hauptteil	10
3 Konfiguration der Vorlage (Main-Funktion)	10
3.1 Persönliche und formale Daten .....	10
3.2 Projektspezifische Daten .....	10
3.3 Unternehmensdaten .....	10
3.4 Dokumentensteuerung .....	11
3.5 Beispiel für den Aufruf .....	11
4 Nutzung der sonstigen Befehle	11
4.1 Abkürzungen .....	11
4.2 Zitieren .....	12
4.3 Codeblöcke .....	12
4.4 Formeln .....	12
4.5 Abbildungen .....	12
4.6 Tabellen .....	13
5 Schluss	14
6 Anhang	15
Literaturverzeichnis	XVI

# **Abbildungsverzeichnis**

Abbildung 1 Hilfe was sehe ich hier .....	13
---	----

# **Abkürzungsverzeichnis**

**LATATW** Long And Tedious Acronym To Write

**SDA** Simply Defined Acronym

**ASDA** Another Simply Defined Acronym

# **Tabellenverzeichnis**

Tabelle 1 DHBW-Betriebszustände .....	13
---------------------------------------	----

# **Quellcodeverzeichnis**

Quellcode 1 Beispielaufruf .....	11
Quellcode 2 lustiger Rust code .....	12

# **Formelverzeichnis**

Gleichung (1) .....	12
---------------------	----

# **Kapitel 1**

## **Einleitung**

# Kapitel 2

## Hauptteil

## Kapitel 3

### Konfiguration der Vorlage (Main-Funktion)

Die Vorlage wird über die Funktion `bericht` konfiguriert. Hier ist eine Übersicht aller Parameter, die an die Funktion übergeben werden können:

#### 3.1 Persönliche und formale Daten

- `Autor`: Der vollständige Name des Verfassers (Content).
- `MatrikelNummer`: Die Matrikelnummer zur eindeutigen Identifikation (String/Content).
- `Kurs`: Die Kursbezeichnung (z. B. „TINF22B1“).
- `Studiengang`: Der Name des Studiengangs (z. B. „Informatik“).

#### 3.2 Projektspezifische Daten

- `Titel`: Das Thema der Arbeit, erscheint groß auf dem Deckblatt.
- `Was`: Die Art der Arbeit (z. B. „Praxisbericht“, „Bachelorarbeit“).
- `Pruefung`: Die Modulbezeichnung oder Prüfungsnummer (z. B. „T3\_2000“).
- `AbgabeDatum`: Das Datum der Einreichung.
- `Dauer`: Der Bearbeitungszeitraum (z. B. „12 Wochen“ oder „560 Stunden“).

#### 3.3 Unternehmensdaten

- `FirmenName`: Name des Ausbildungsbetriebs.
- `FirmenStadt`: Standort des Betriebs (wird für die Erklärung benötigt).
- `Firmenlogo`: Das Logo der Firma. Einbindung via `image("pfad/zu/logo.png")`.
- `BetreuerFirma`: Name des Ansprechpartners im Betrieb.
- `BetreuerDHBW`: Name des betreuenden Dozenten oder Gutachters der DHBW.

## 3.4 Dokumentensteuerung

- **Sperrvermerk:** Ein boolescher Wert (`true` oder `false`). Bei `true` wird automatisch ein Sperrvermerk in die Erklärung eingefügt.
- **Zusammenfassung:** Der Inhalt des Abstracts. Am besten via `include "content/abstract.typ"` einbinden.
- **acronyms:** Ein Dictionary oder eine YAML-Datei (`yaml("abk.yml")`) mit Abkürzungen für das automatische Verzeichnis.
- **bibliography-content:** Die Literaturquelle, eingebunden über die `bibliography()`-Funktion.

## 3.5 Beispiel für den Aufruf

Ein typischer Aufruf in der `main.typ` sieht wie folgt aus:

```

1 #import "@local/turbocharged-dhw:0.1.0" : bericht
2
3 #show: bericht.with(
4   Autor: "Rudi Regentonne",
5   Titel: "Analyse des Kaffeeverbrauchs im Homeoffice",
6   Studiengang: "Angewandte Praktikantenwissenschaften",
7   Firmenlogo: image("assets/firma-logo.png"),
8   Sperrvermerk: true,
9   Zusammenfassung: include "content/abstract.typ",
10  acronyms: yaml("abk.yml"),
11 )

```

Quellcode 1: Beispielaufruf

# Kapitel 4

## Nutzung der sonstigen Befehle

### 4.1 Abkürzungen

- `#init-acronyms(dict)`: Initialisiert die Abkürzungen.
- `#acr(key)` (oder `#ac`): Zeigt Abkürzung an (beim ersten Mal voll, danach kurz).
- `#acrpl(key)` (oder `#acp`): Plural-Version von `#acr`.
- `#acrcap(key)`: Wie `#acr`, aber Definition wird großgeschrieben.
- `#acrfull(key)` (oder `#acf`): Zeigt immer die Langform.
- `#acs(key)`: Zeigt immer nur die Kurzform.
- `#acl(key)`: Zeigt nur die Definition (Langform) an.
- `#print-index()`: Erzeugt das Abkürzungsverzeichnis.
- `#reset-all-acronyms()`: Setzt alle Abkürzungen zurück.

- `#acused(key)`: Markiert Abkürzung als „verwendet“, ohne Text zu drucken.

## 4.2 Zitieren

Geht mit `@bibname` oder mit `#cite(<bibname>)@`

## 4.3 Codeblöcke

Inline kann mit einfachen Backticks eingebunden werden `Inlinocode`. Für Codeblöcke kann entweder ```Sprache code``` oder

```
#import "@local/turbocharged-dhbw:0.1.0": code
#code(
    firstnumber: 1,
    stepnumber: 1,
    numbers: true,
    caption: "lustiger Rust code",
    highlight: ( 2, 4),
)[``rust
fn main() {
    let semester_planung = vec![None; 3]; // Drei Jahre voller Leere
    let klausur_status = "abgegeben".repeat(100);
    let korrektur_dauer = std::time::Duration::from_secs(u64::MAX);
}
```]
```

Heraus kommt dann:

```
1 fn main() {
2     let semester_planung = vec![None; 3]; // Drei Jahre voller Leere
3     let klausur_status = "abgegeben".repeat(100);
4     let korrektur_dauer = std::time::Duration::from_secs(u64::MAX);
5 }
```

Quellcode 2: lustiger Rust code

## 4.4 Formeln

`$1 + $1$` gibt eine inline Formel:  $1 + 1$ . Um eine Formel zu schreiben, die auch im Formelverzeichnis auftaucht wird dieser Syntax verwendet: `

```
#figure(
    $ y = m dot x + c $,
    kind: math.equation,
)
```

Das Ergebnis:

$$y = m \cdot x + c \quad (1)$$

## 4.5 Abbildungen

Bilder werden so eingefügt:

```
#figure(
  image("../assets/dhbw-logo.png", width: 40%),
  caption: [
    Hilfe was sehe ich hier
  ],
)
```



Abbildung 1: Hilfe was sehe ich hier

## 4.6 Tabellen

Tabellen, die auch im Tabellenverzeichnis landen sollen, werden so erstellt:

```
#figure(
  table(
    columns: (1fr, 1fr),
    fill: (col, row) => if row == 0 { maroon.lighten(80%) },
    [*Vorgang*], [*Status*],
    [Korrektur], [Wie bitte?],
    [Moodle], [Wartungsarbeiten],
    [Dualis-Server], [Glasfaser angebaggert],
    [Digitaltechnik Klausur], [geschrieben (1998)],
    [Exmatrikulation], [ausgefüllt],
  ),
  caption: [DHBW-Betriebszustände],
)
```

Vorgang	Status
Korrektur	Wie bitte?
Moodle	Wartungsarbeiten
Dualis-Server	Glasfaser angebaggert
Digitaltechnik Klausur	geschrieben (1998)
Exmatrikulation	ausgefüllt

Tabelle 1: DHBW-Betriebszustände

# **Kapitel 5**

## **Schluss**

# **Kapitel 6**

## **Anhang**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et. [1], [2], [3], [4], [5], [6], [7], [8], [9]

# Literaturverzeichnis

- [1] „Single Sign-on (SSO): Ein Passwort, sie alle zu knechten, sie alle zu finden... und hoffentlich nie gehackt zu werden“. Zugegriffen: 23. September 2025. [Online]. Verfügbar unter: <https://auth0.com/de/intro-to-iam/what-is-single-sign-on-sso>
- [2] „Das große WLAN-Orakel: Warum dein Internet im 2.4 GHz Band stirbt, sobald die Mikrowelle angeht (IEEE 802.11b-1999)“, *IEEE - Institut für extrem komplizierte Abkürzungen*, S. 1–96, 2000.
- [3] „Arduino - Wie man 40 Kabel für zwei Zeilen Text verschwendet | Ein Tutorial für Masochisten“. Zugegriffen: 25. September 2025. [Online]. Verfügbar unter: <https://arduinogetstarted.com/tutorials/arduino-lcd>
- [4] „MFRC522: Die Kunst, Karten zu lesen, ohne sie physisch zu berühren (Magie für Anfänger)“, Bd. 2016, 2016.
- [5] „ASP.NET Core: Microsofts Versuch, Java-Entwickler mit glänzenden neuen Spielzeugen zu ködern“. Zugegriffen: 23. September 2025. [Online]. Verfügbar unter: <https://learn.microsoft.com/de-de/aspnet/core/overview?view=aspnetcore-9.0>
- [6] „PCI Express: Ein Stammbaum von Steckplätzen, die immer schneller werden, während meine Grafikkarte immer fetter wird“. Zugegriffen: 15. September 2025. [Online]. Verfügbar unter: <https://www.elektronik-kompendium.de/sites/com/0904051.htm>
- [7] D. rote Hut (Red Hat), „Was ist eine API? – Oder: Wie Software-Komponenten miteinander flirten, ohne sich zu kennen“. Zugegriffen: 11. September 2025. [Online]. Verfügbar unter: <https://www.redhat.com/de/topics/api/what-are-application-programming-interfaces>
- [8] R. Hat, „REST APIs: Endlich mal eine Pause (REST) vom SOAP-Wahnsinn“. Zugegriffen: 15. September 2025. [Online]. Verfügbar unter: <https://www.redhat.com/de/topics/api/what-is-a-rest-api>
- [9] „Was ist ein Mikrocontroller? Ein Computer für Ameisen, der in meiner Kaffeemaschine lebt“. Zugegriffen: 15. September 2025. [Online]. Verfügbar unter: <https://heinen-elektronik.de/glossar/mikrocontroller/>