



## WSL-2: Der Warp-Antrieb für Windows-Entwickler:innen

Unter Windows entwickeln und in Unix kompilieren

Mirko Pleli | Sybit GmbH

13.07.2023

[excellence in customer xperience]

# Über mich

- Mirko Pleli
- Solution Developer
- Radolfzell am Bodensee
- Java, Spring-Boot, Nest.js, Angular und Microsoft Azure



Performancesteigerung, Kosteneinsparung und Open-Source?

# Agenda

- ✓ Entwicklung im Wandel
- ✓ WSL-2 vs. WSL-2 (GUI)
- ✓ Performancesteigerung
- ✓ Kosteneinsparung
- ✓ Open-Source

# Entwicklung im Wandel

## Die Herausforderungen

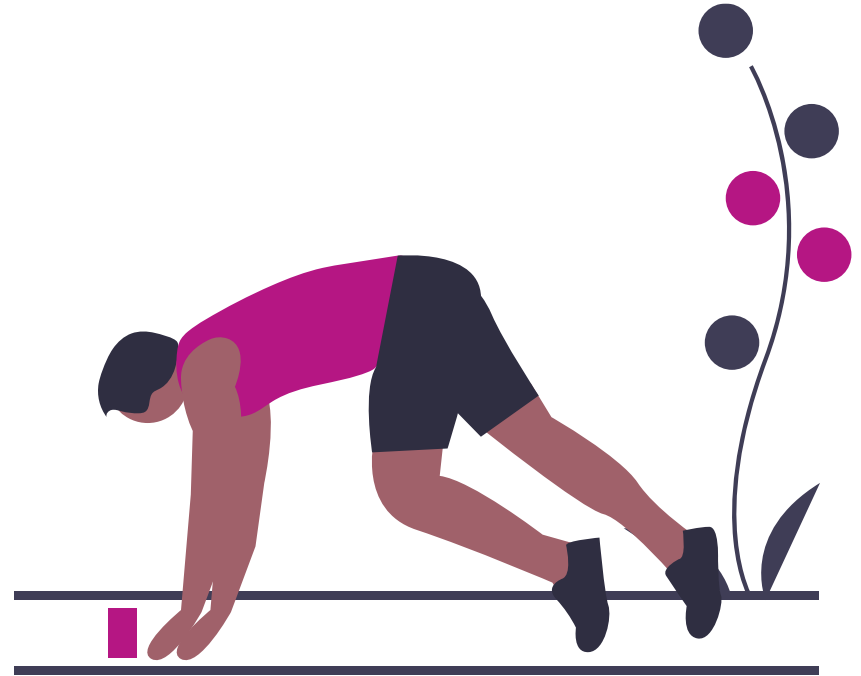
- Performance Einbuße in Commerce Projekten
- Kostspieliger Einsatz von JRrebel
- NTFS stellt sich als Bottle-Neck heraus
- VM-Setup erhöht den administrativen Aufwand
- MacBooks „nicht“ integrierbar in die aktuelle IT-Infrastruktur



# Entwicklung im Wandel

## WSL-2 (Windows Subsystem für Linux)

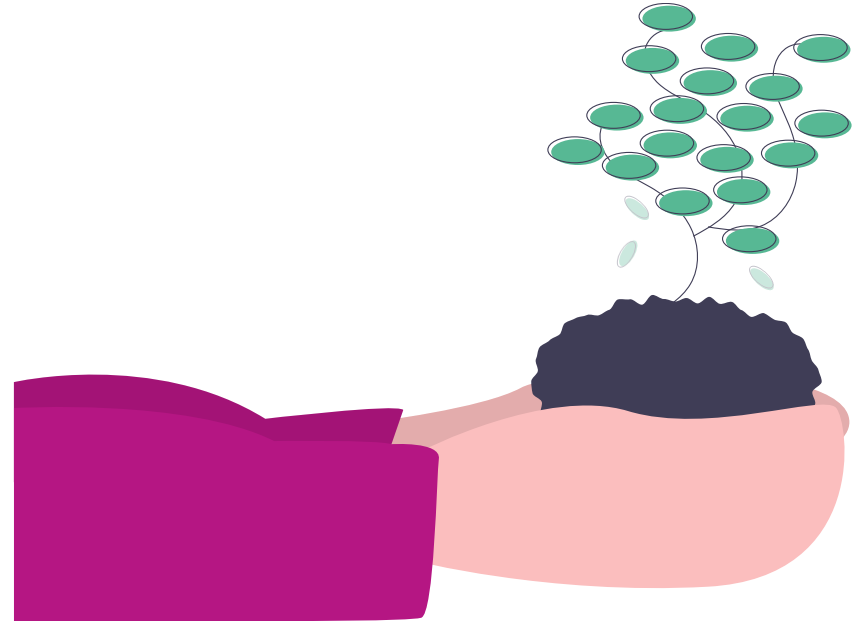
- Veröffentlichung der WSL-2 im Frühjahr 2020
- Eigenständiger Linux-Kernel
- Zugriff auf die Distribution über den Windows-Explorer
- 87% gegenüber nativer Linux-Leistung
- WSL-2 GUI in der Pipeline



# Entwicklung im Wandel

## Die Out of the Box Lösung

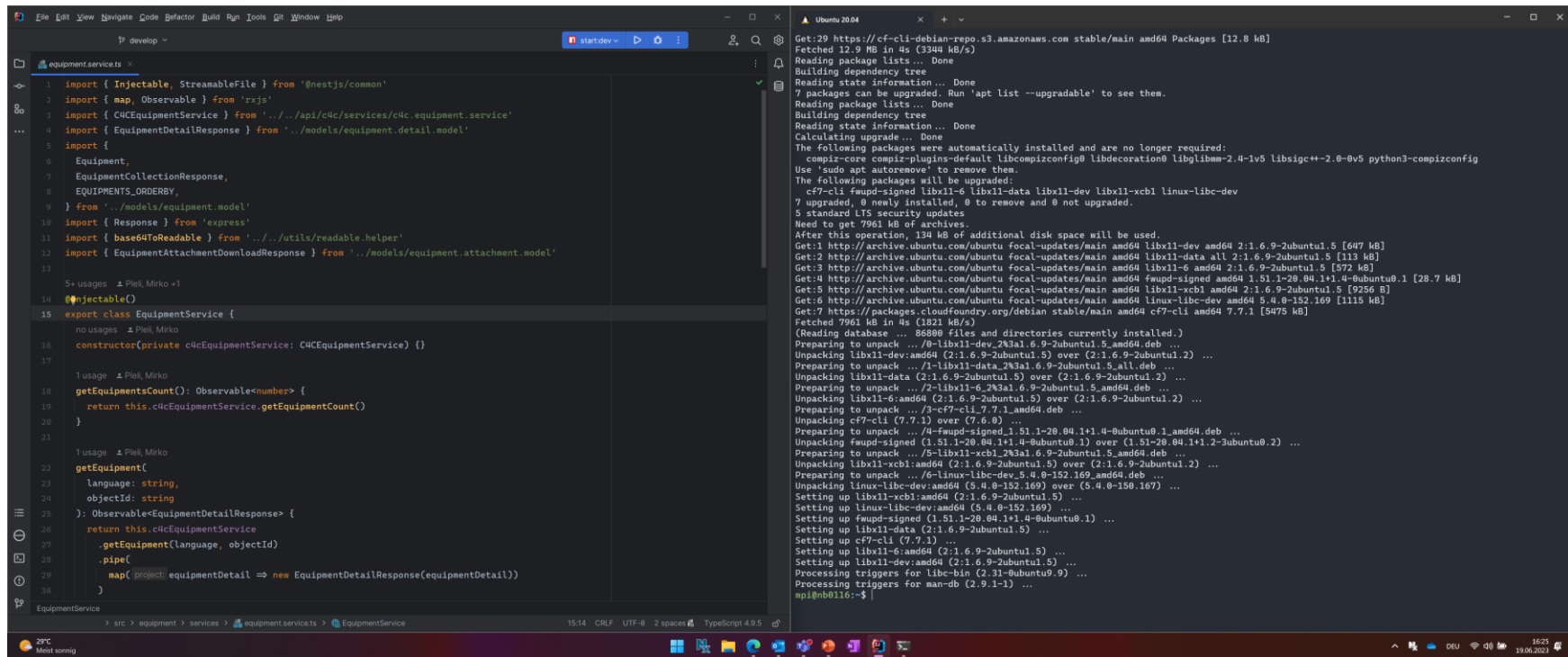
- Installation über den Windows Store
- Aktualisierung über die Windows Updates
- Keine Freigabe von MAC-Adressen notwendig
- Keine zusätzlichen Kosten und Lizenzen
- Visual Studio Code und IntelliJ IDEA stellen Integrationen bereit



WSL-2 vs. WSL-2 (GUI)?

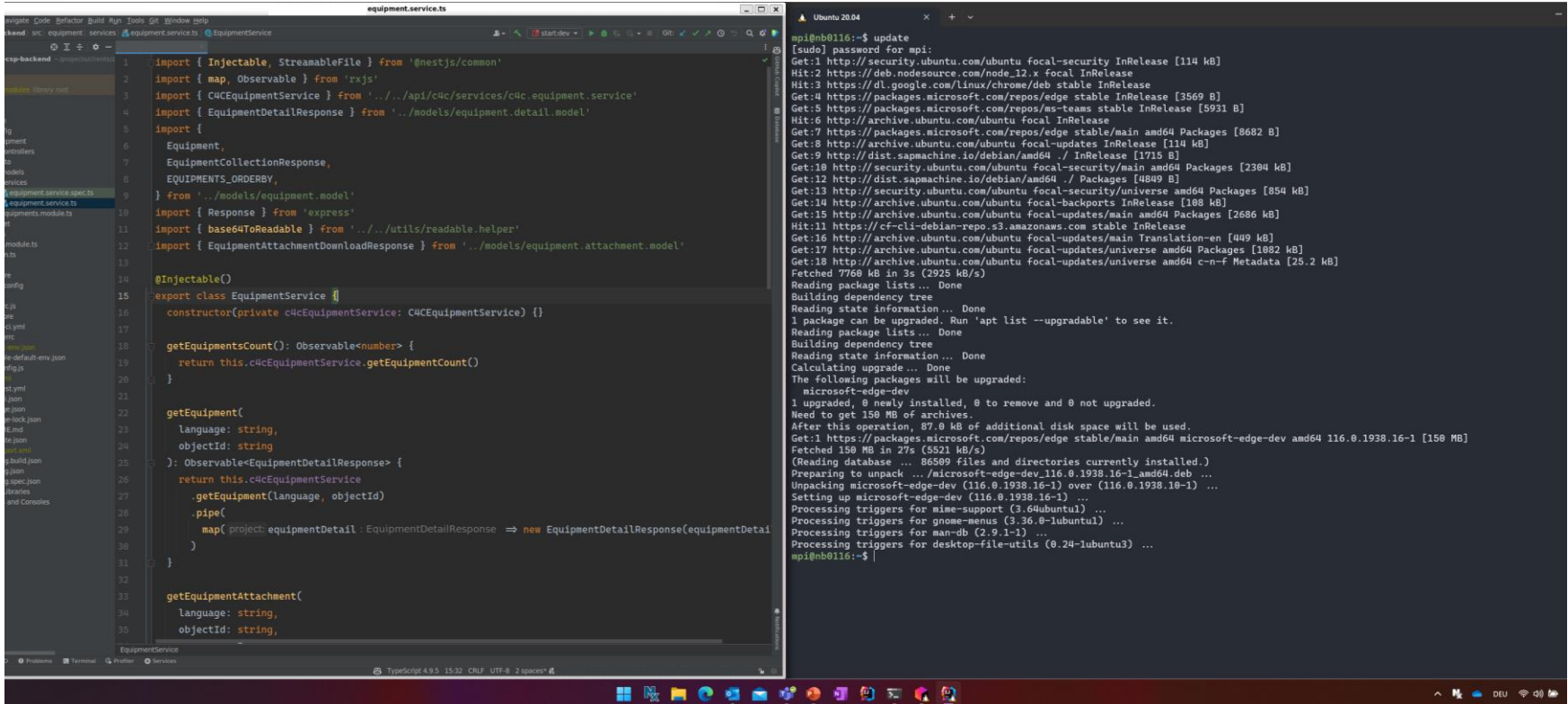


## Unter Windows entwickeln und in Ubuntu kompilieren



# WSL-2 vs. WSL-2 (GUI)

Unter Ubuntu entwickeln und in Ubuntu kompilieren in Windows



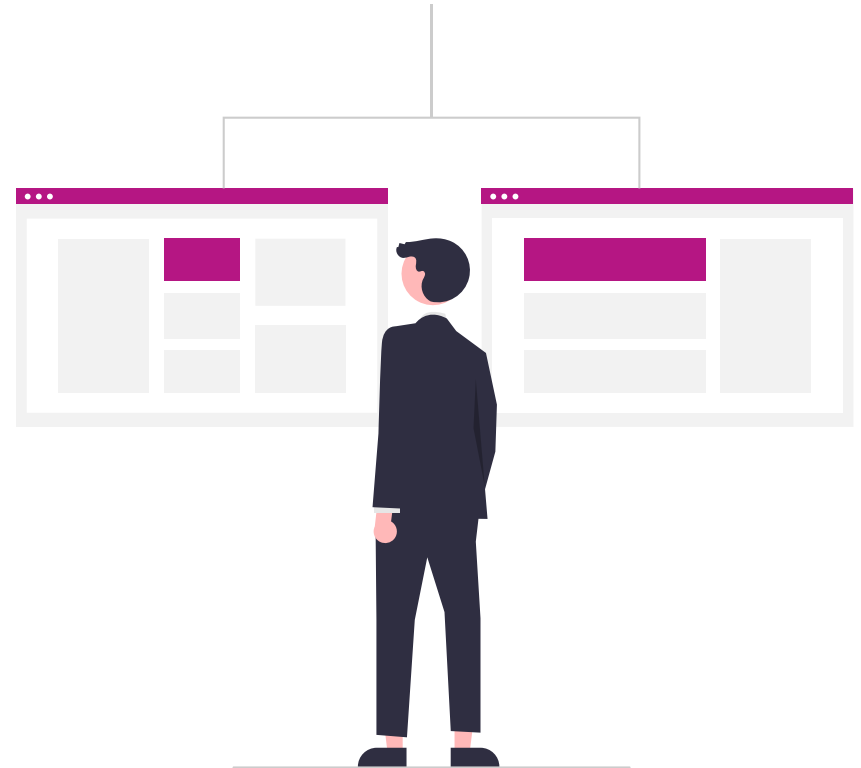
```
equipment.service.ts
1 import { Injectable, StreamableFile } from '@nestjs/common'
2 import { map, observable } from 'rxjs'
3 import { C4CEquipmentService } from '../api/c4c/services/c4c.equipment.service'
4 import { EquipmentDetailResponse } from '../models/equipment.detail.model'
5 import {
6   Equipment,
7   EquipmentCollectionResponse,
8   EQUIPMENTS_ORDERBY,
9 } from '../models/equipment.model'
10 import { Response } from 'express'
11 import { base64ToReadable } from '../utils/readable.helper'
12 import { EquipmentAttachmentDownloadResponse } from '../models/equipment.attachment.model'
13
14 @Injectable()
15 export class EquipmentService {
16   constructor(private c4cEquipmentService: C4CEquipmentService) {}
17
18   getEquipmentsCount(): Observable<number> {
19     return this.c4cEquipmentService.getEquipmentCount()
20   }
21
22   getEquipment(
23     language: string,
24     objectId: string
25 ): Observable<EquipmentDetailResponse> {
26     return this.c4cEquipmentService
27       .getEquipment(language, objectId)
28       .pipe(
29         map(project: EquipmentDetail : EquipmentDetailResponse => new EquipmentDetailResponse(EquipmentDetail
30       ))
31   }
32
33   getEquipmentAttachment(
34     language: string,
35     objectId: string,
```

```
mp@nb0116:~$ update
[sudo] password for mpi:
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Hit:2 https://deb.nodesource.com/node_12.x focal InRelease
Hit:3 https://dl.google.com/linux/chrome/deb stable InRelease
Get:4 https://packages.microsoft.com/repos/edge stable InRelease [3569 B]
Get:5 https://packages.microsoft.com/repos/ms-teams stable InRelease [5931 B]
Hit:6 http://archive.ubuntu.com/ubuntu focal InRelease
Get:7 https://packages.microsoft.com/repos/edge stable/main amd64 Packages [8682 B]
Get:8 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:9 http://dist.sagemath.org/debian/amd64 ./ InRelease [1715 B]
Get:10 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [2304 kB]
Get:11 http://dist.sagemath.org/debian/amd64 ./ Packages [4849 B]
Get:12 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [854 kB]
Get:13 http://archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:14 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [2686 kB]
Hit:15 https://cf-cli-debian-repo.s3.amazonaws.com stable InRelease
Get:16 http://archive.ubuntu.com/ubuntu focal-updates/main Translation-en [449 kB]
Get:17 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1882 kB]
Get:18 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 c-n-f Metadata [25.2 kB]
Fetched 7768 kB in 3s (2925 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  microsoft-edge-dev
1 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 150 MB of archives.
After this operation, 87.0 kB of additional disk space will be used.
Get:1 https://packages.microsoft.com/repos/edge stable/main amd64 microsoft-edge-dev amd64 116.0.1938.16-1 [150 MB]
Fetched 150 MB in 27s (5521 kB/s)
(Reading database ... 86509 files and directories currently installed.)
Preparing to unpack .../microsoft-edge-dev-116.0.1938.16-1_amd64.deb ...
Unpacking microsoft-edge-dev (116.0.1938.16-1) over (116.0.1938.10-1) ...
Setting up microsoft-edge-dev (116.0.1938.16-1) ...
Processing triggers for mime-support (3.64ubuntu1) ...
Processing triggers for gnome-menus (3.36.0-1ubuntu1) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for desktop-file-utils (0.24-lubuntu3) ...
mp@nb0116:~$
```

# WSL-2 vs. WSL-2 (GUI)

## Vor und Nachteile

- Vorteile
  - Natives arbeiten direkt in Windows möglich
  - Indexierung nach Build performanter
- Nachteile
  - Ausführen von Cypress-Tests nicht direkt möglich
  - Hohe Beanspruchung von Arbeitsspeicher
  - Keine nativen Fenster wie unter Windows



Performancesteigerung?

# Performancesteigerung

SAP Commerce (Java, Spring-Boot, Solr...)

Gerät	Initialize	Build	Clean Build	Starten
DELL Precision 7550 OS: Win10	30 Min.	4-5 Min.	17 Min.	5-6 Min.
WSL-2: Ubuntu 20.04	8 Min.	1-2 Min.	2-3 Min.	1-2 Min.

# Performancesteigerung

SAP Spartacus Storefront (Angular...)

Gerät	Run 1	Run 2	Run 3	Run 4
DELL Precision 7490: OS: Win10	80 Sek.	66 Sek.	61 Sek.	64 Sek.
WSL-2: Ubuntu 20.04	26 Sek.	24 Sek.	25 Sek.	24 Sek.

Kosteneinsparung?

# Kosteneinsparung

## Unter Windows

- Je Arbeitstag fallen 10 bis 15 Builds / Starts an
- 10 Builds \* 5 Min. = 50 Min. Wartezeit
- 10 Starts \* 5 Min. = 50 Min. Wartezeit
- Entspricht einer Wartezeit von 100 Min. unter Windows

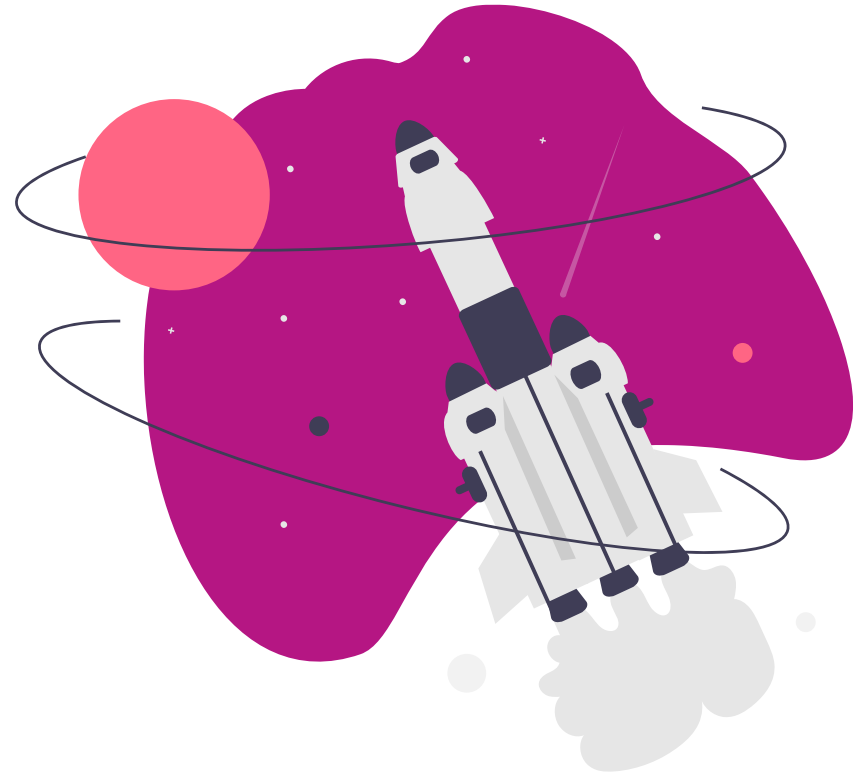




# Kosteneinsparung

Unter Windows mit WSL-2

- 10 Builds \* 2 Min. = 20 Min. Wartezeit
- 10 Starts \* 2 Min. = 20 Min. Wartezeit
- Entspricht einer Wartezeit von 40 Min. unter Windows mit der WSL-2



Open-Source?

# Open-Source Quick Start Guide



Fragen?



## Sybit GmbH

St.-Johannis-Str. 1-5  
78315 Radolfzell  
mirko.pleli@sybit.de

