

[google.com](https://www.google.com)

mit intellij ein war-file für das Deployment auf einem Tomcat per Manager-Site erstellen, und welche Vorbereitungen bei einem SpringBoot projekt dafür nötig sind, wenn dieses bereits lokal auf dem integrierten Tomcat laufen kann.

10–12 Minuten

Neue Suche starten

KI-Modus-Verlauf

mit intellij ein war-file für das Deployment auf einem Tomcat per Manager-Site erstellen, und welche Vorbereitungen bei einem SpringBoot projekt dafür nötig sind, wenn dieses bereits lokal auf dem integrierten Tomcat laufen kann.

Um ein Spring Boot Projekt für das Deployment als WAR-File auf einem externen Tomcat (via Manager-App) vorzubereiten, sind Anpassungen am Code und an der Build-Konfiguration in IntelliJ erforderlich.

1. Vorbereitungen im Spring Boot Projekt

Standardmäßig erstellt Spring Boot ein ausführbares JAR mit

eingebettetem Tomcat. Für ein externes Deployment müssen Sie dies umstellen:

- **Verpackungstyp ändern:** Stellen Sie in der `pom.xml` (Maven) den Typ auf WAR um:
`<packaging>war</packaging>`.
- **Hauptklasse anpassen:** Ihre Hauptklasse muss `SpringBootServletInitializer` erweitern und die Methode `configure` überschreiben. Dies ermöglicht es dem externen Tomcat, die Anwendung zu starten:

java

```
@SpringBootApplication
public class MyWebApp extends
SpringBootServletInitializer {
    @Override
    protected SpringApplicationBuilder
configure(SpringApplicationBuilder application) {
        return application.sources(MyWebApp.class);
    }
    public static void main(String[] args) {
        SpringApplication.run(MyWebApp.class, args);
    }
}
```

Verwende Code mit Vorsicht.

- **Eingebetteten Tomcat auf "provided" setzen:** Damit der integrierte Tomcat nicht mit dem externen Server kollidiert, markieren Sie die Abhängigkeit als provided:

xml

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-tomcat</
artifactId>
  <scope>provided</scope>
</dependency>
```

Verwende Code mit Vorsicht.

2. Erstellen des WAR-Files in IntelliJ

Nachdem die Konfiguration steht, können Sie die Datei generieren:

1. Öffnen Sie das **Maven-Tool-Fenster** in IntelliJ (rechts am Rand).
2. Navigieren Sie zu Lifecycle und führen Sie nacheinander **clean** und **package** aus.
3. Das fertige WAR-File finden Sie anschließend im Projektordner unter dem Verzeichnis **/target**.

3. Deployment über die Tomcat Manager-Site

Sobald Sie das WAR-File haben, können Sie es hochladen:

1. Starten Sie Ihren externen Tomcat-Server und rufen Sie die **Manager App** auf (meist unter `http://localhost:8080/manager/html`).
2. Scrollen Sie zum Bereich **"WAR file to deploy"**.
3. Wählen Sie über "Datei auswählen" Ihr WAR-File aus dem **/target**-Ordner Ihres Projekts aus.
4. Klicken Sie auf **"Deploy"**. Die Anwendung wird entpackt und unter

einem Pfad erreichbar, der dem Namen des WAR-Files entspricht (z. B. /mein-projekt-0.0.1-SNAPSHOT).

Wichtiger Hinweis: Der Context-Pfad (die URL Ihrer App) wird beim Manager-Deployment standardmäßig durch den Dateinamen des WARs bestimmt. Wenn die App direkt unter / (Root) laufen soll, benennen Sie das File vor dem Hochladen in ROOT.war um.

Öffentlicher Link wird erstellt...

Öffentlicher Link wird erstellt...

- Easily Deploy to Tomcat Webapps Folder?

15.12.2012 — If you did not do any of those things, then you need to use the URL `http://localhost:8080/NameOfWarFile/TestServlet`
In IDEA, in a ...

JetBrains

- Deploy a Spring Boot WAR into a Tomcat Server | Baeldung

08.01.2024 — To build our Tomcat-deployable WAR application, we'll execute the `mvn clean package`. After that, our WAR file is generated at `target/`...

Baeldung



- Demystifying Tomcat, WAR & JAR: A Modern Guide to Spring Boot ...

06.06.2025 — Deploying Spring Boot WAR to a Local or Production Tomcat * Step 1: Install Tomcat. Download from `https://tomcat.apache.org`. Extra...

Medium



Alle anzeigen

lässt sich das ggf so aufbauen, dass es per Schalter umstellbar ist, ob der IntelliJ-interne Tomcat genutzt werden soll, oder ein Far-File für einen externen Tomcat produziert wird?

Ja, das lässt sich sehr elegant über

Maven-Profile lösen. So können Sie zwischen dem Standard-Modus (JAR mit eingebettetem Tomcat für die lokale Entwicklung) und dem Deployment-Modus (WAR für externen Tomcat) umschalten.

1. Maven Profile in der pom.xml definieren

Ersetzen Sie die statische `<packaging>`-Angabe und die Tomcat-Dependency durch ein Profil-Konstrukt:

xml

```
<!-- Standardmäßig als JAR verpacken für lokale
Entwicklung -->
<packaging>${packaging.type}</packaging>

<properties>
    <packaging.type>jar</packaging.type>
</properties>

<profiles>
    <profile>
```

```
<id>external-tomcat</id>
<properties>
    <packaging.type>war</packaging.type>
</properties>
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</
groupId>
        <artifactId>spring-boot-starter-
tomcat</artifactId>
        <scope>provided</scope>
    </dependency>
</dependencies>
</profile>
</profiles>
```

Verwende Code mit Vorsicht.

2. Code-Anpassung (funktioniert für beides)

Die Anpassung der Hauptklasse (Erweiterung von `SpringBootServletInitializer`), wie in der ersten Antwort beschrieben, ist **abwärtskompatibel**. Sie schadet nicht, wenn Sie die Anwendung lokal als JAR starten. Sie können diese Klasse also dauerhaft so lassen:

java

```
@SpringBootApplication
public class MyWebApp extends
SpringBootServletInitializer {
    // ... configure-Methode wie oben beschrieben ...
}
```

Verwende Code mit Vorsicht.

3. Umschalten in IntelliJ

In der IntelliJ-Oberfläche können Sie nun bequem zwischen den Modi wechseln:

1. Öffnen Sie rechts das **Maven-Fenster**.
2. Klappen Sie den Reiter **Profiles** auf.
3. **Für lokales Ausführen (interner Tomcat):** Deaktivieren Sie das Häkchen bei `external-tomcat`. Führen Sie die App einfach über den "Run"-Button von IntelliJ aus.
4. **Für das Deployment (externer Tomcat):**
 - Setzen Sie das Häkchen bei `external-tomcat`.
 - Klicken Sie im Maven-Fenster unter Lifecycle auf `clean` und dann auf `package`.
 - IntelliJ nutzt nun die Profil-Einstellungen, setzt den Tomcat auf `provided` und erstellt ein `.war` im `target`-Ordner.

4. Besonderheit beim lokalen Starten

Wenn Sie das Profil `external-tomcat` in IntelliJ **aktiviert** haben, wird der interne Tomcat als `provided` markiert. Wenn Sie dann versuchen, die Anwendung über den grünen "Play"-Button in IntelliJ zu starten, fehlen die Tomcat-Klassen im Classpath.

Lösung: Sorgen Sie dafür, dass beim normalen Ausführen in IntelliJ die Option **"Include dependencies with 'Provided' scope"** in der *Run/Debug Configuration* aktiviert ist (unter "Modify options" zu finden), oder schalten Sie das Profil einfach kurz aus, bevor Sie lokal starten.

Öffentlicher Link wird erstellt...

Öffentlicher Link wird erstellt...

Alle anzeigen

ist die Sache mit einer anderen IDE ggf komfortabler lösbar?

Ich überlege

Ergebnisse werden gesucht

In der Java-Welt gibt es keine IDE, die das Umschalten zwischen JAR (intern) und WAR (extern) grundlegend "einfacher" löst als IntelliJ, da dies eine strukturelle Änderung am Build-Prozess (Maven/Gradle) und am Code ist. Dennoch bieten andere IDEs unterschiedliche Komfort-Ansätze:

1. Spring Tool Suite (STS) / Eclipse

STS ist eine auf Eclipse basierende IDE, die speziell auf Spring optimiert ist.

- **Vorteil:** Sie verfügt über eine sehr tiefgehende Integration des "**Boot Dashboards**". Dort können Sie Anwendungen oft mit einem Klick auf verschiedene Zielumgebungen pushen.
- **Nachteil:** Der klassische "WAR Export"-Wizard wurde in neueren STS-Versionen (STS4) standardmäßig entfernt, da der Fokus fast nur noch auf Spring Boot (JAR) liegt. Man muss oft zusätzliche Plugins (m2e-wtp) manuell nachinstallieren, um den alten Komfort für WAR-Files zurückzubekommen.

2. Apache NetBeans

NetBeans gilt traditionell als die IDE mit der stärksten Unterstützung für klassische Application Server wie Tomcat.

- **Vorteil:** In NetBeans 26 (Stand 2026) ist die Verwaltung externer Server sehr intuitiv in die Projektstruktur integriert. Wenn Sie ein

Projekt als "Web Application" anlegen, übernimmt NetBeans viele Deployment-Schritte im Hintergrund.

- **Nachteil:** Bei Spring Boot Projekten zwingt NetBeans Ihnen dennoch oft das Maven-Modell auf, sodass Sie am Ende wieder bei den gleichen `pom.xml`-Anpassungen landen wie in IntelliJ.

3. VS Code

Mit den **Spring Boot Extension Packs** ist VS Code eine sehr leichtgewichtige Alternative.

- **Komfort:** Das Umschalten von Profilen (wie dem `external-tomcat`-Profil) erfolgt hier meist über einfache Klicks in der Statusleiste oder in der Maven-Ansicht.
- **Einschränkung:** VS Code bietet weniger visuelle Unterstützung für die Verwaltung der Tomcat-Manager-Site selbst; hier müssen Sie meist manuell im Browser agieren.

Fazit: Was ist am komfortabelsten?

Der "Schalter"-Ansatz über **Maven-Profile** (wie in der vorherigen Antwort beschrieben) bleibt die professionellste und IDE-unabhängigste Lösung.

- **IntelliJ Ultimate** bietet den höchsten Komfort, da es die **"Application Servers"**-Ansicht besitzt. Dort können Sie Ihren externen Tomcat hinterlegen und IntelliJ so konfigurieren, dass es das WAR-File nach dem Build automatisch per Mausklick auf den Server lädt, ohne dass Sie die Manager-Website im Browser öffnen müssen.
- Wenn Sie eine **kostenlose** Alternative suchen, die sich stärker wie eine klassische "Server-zentrierte" IDE anfühlt, ist **NetBeans** einen Blick wert.

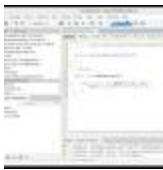
Öffentlicher Link wird erstellt...

Öffentlicher Link wird erstellt...

- NetBeans 26 Added Java SE 25 Support for Tomcat, TomEE ...

21.05.2025 — NetBeans 26 Added Java SE 25 Support for Tomcat, TomEE & GlassFish | UbuntuHandbook. NetBeans 26 Added Java SE 25 Support for Tomc...

UbuntuHandbook



- Java IDE: prefer Eclipse / IntelliJ for Java | by Alice Dai - Medium

28.09.2025 — Excellent Remote - SSH /Containers support for editing code inside containers or remote machines. Simplicity. Lower memory footpri...

Medium

- Export WAR is missing in STS4 · Issue #53 - GitHub

28.04.2018 — martinlippert commented. martinlippert. on May 29, 2018. Member. The WAR export feature is part of the wider Java EE and web featu...

GitHub



Alle anzeigen

Schließen

Mehr Eingabeoptionen

Mikrofon

Bild hochladen

Senden