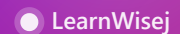


Konfiguration, Sicherheit und Deployment

Konfiguration und Secrets · Sicherheit und Deployment-Reife



Konfiguration und Secrets

Welche Konfigurationsdateien Sie vor einem Release prüfen, wie Sie Debug- und Produktiveinstellungen unterscheiden und wie Sie Lizenzschlüssel und Secrets sicher handhaben.



Lernziel der Lektion

Wissen Sie, welche Konfigurationsdateien Sie vor dem Release prüfen müssen, unterscheiden Sie Debug, Staging und Produktivbetrieb, und gehen Sie sicher mit License Keys, Connection Strings und anderen Secrets um.

1. Zu prüfende Konfigurationsdateien

In einem Wisej.NET-Projekt kann die Konfiguration an mehreren Stellen liegen. Sie müssen nicht jede fortgeschrittene Einstellung auswendig kennen – aber Sie müssen wissen, wo Sie vor dem Deployment nachsehen.

Bereich	Was zu prüfen ist	Warum es wichtig ist
<code>Program.cs</code> / Start	Welches Fenster bzw. welche Seite startet, umgebungsspezifische Startlogik, Einrichtung der Services.	Die bereitgestellte App muss den richtigen Einstiegspunkt öffnen.
<code>Web.config</code> / Hosting-Konfiguration	Debug-Flag, Handler, Servereinstellungen, Rewrite-Regeln, Fehlerverhalten.	Das Hosting unter IIS hängt stark von einer korrekten Konfiguration ab.
<code>Default.json</code> / App-JSON	Wisej.NET-Endpunkt, Theme, URL, Fenster-/Seiteneinstellungen, Verweise auf statische Ressourcen.	Falsche URLs oder Themes können dazu führen, dass eine bereitgestellte App nicht startet oder fehlerhaft aussieht.
Themes & statische Dateien	Theme-Ordner, CSS, Bilder, Icons, JavaScript-Dateien.	Die Ressourcen müssen im Deployment-Paket enthalten sein.
Logging-Konfiguration	Log-Ziel, Log-Level, geschützte Fehlerdetails.	Im Produktivbetrieb braucht man Fehlersuche, ohne Interna preiszugeben.

2. Debug, Staging und Produktivbetrieb

Debug-Einstellungen helfen während der Entwicklung, doch Produktiveinstellungen sollten sicherer und kontrollierter sein. Prüfen Sie vor dem Release die Umgebung und stellen Sie sicher, dass kein Debug-Verhalten versehentlich aktiv bleibt.

- **Debug** – detaillierte Fehler, lokales Testen, einfachere Fehlersuche; nicht sicher für öffentliche Benutzer.
- **Staging** – ein praxisnaher Testbereich, der vor dem Release genutzt wird.
- **Produktivbetrieb** – sichere Fehlermeldungen, geschützte Secrets, stabile Konfiguration, aktiviertes Logging.



Regel für ein sicheres Release

Benutzer sollten eine hilfreiche Meldung sehen; Entwickler sehen die technischen Details in sicheren serverseitigen Logs. Zeigen Sie in der Oberfläche niemals Stack Traces, Dateipfade, License Keys oder Connection Strings an.

3. Umgang mit Secrets und License Keys

Sensible Werte sollten nicht fest in UI-Code, Button-Events, Screenshots oder die Versionsverwaltung geschrieben werden – License Keys, Datenbank-Connection-Strings, API-Keys, SMTP-Passwörter und Zugangsdaten für das Deployment.

Unsichere Praxis	Sicherere Praxis
Einen License Key fest in eine Seite oder ein Button-Event schreiben.	Eine sichere Konfiguration oder Umgebungsvariablen verwenden.
Connection Strings in Git einchecken.	Secrets außerhalb der Versionsverwaltung halten; Deployment-Secrets verwenden.
Rohe Exception-Details für Benutzer anzeigen.	Details serverseitig protokollieren und eine sichere Meldung anzeigen.
Zugangsdaten für den Produktivbetrieb in Screenshots weitergeben.	Secrets in Schulungsmaterialien schwärzen oder gar nicht erst zeigen.



Sie sind bereit für die nächste Lektüre

Sie können die wichtigsten Konfigurationsdateien finden, Debug vom Produktivbetrieb trennen und Secrets schützen. Als Nächstes: Authentifizierung und Autorisierung, Logging und eine Deployment-Checkliste.

Sicherheit und Deployment-Reife

Authentifizierung und Autorisierung auf Einsteigerniveau, sicheres Logging, Deployment-Ziele und eine verbindliche Release-Checkliste.



Lernziel der Lektion

Verstehen Sie den Unterschied zwischen Authentifizierung und Autorisierung auf Junior-Niveau, protokollieren Sie sicher und erstellen Sie eine Deployment-Checkliste für IIS, Kestrel oder Cloud-Hosting.

1. Authentifizierung & Autorisierung

Authentifizierung beantwortet die Frage *Wer ist der Benutzer?* **Autorisierung** beantwortet die Frage *Was darf dieser Benutzer tun?* Modul 9 führt dies als Entwurfskonzept ein, noch vor einer vollständigen Identitätsimplementierung.

- **Authentifizierung:** Anmeldung, Identity Provider, Session-Identität, Benutzername, Rollen-Claims.
- **Autorisierung:** Rollenprüfungen, Zugriff auf Seiten, Zugriff auf Befehle, ausgeblendete oder deaktivierte Buttons.
- **Beispiel:** Ein Support Agent darf Tickets verwalten; ein Admin darf Deployment-Einstellungen bearbeiten.
- Verlassen Sie sich niemals *nur* auf das Ausblenden eines Buttons – auch die serverseitige Logik sollte die Aktion schützen.

```
bool canReviewDeployment =
    currentUser.Role == "Admin" || currentUser.Role == "Team Lead";

btnReviewPackage.Enabled = canReviewDeployment;
lblPermission.Text = canReviewDeployment
    ? "Deployment review available."
    : "Review is restricted.";
```

2. Checkliste für Logging und Fehlersuche

Logging sollte festhalten, was passiert ist, ohne Secrets preiszugeben. Ein gutes Log für die Fehlersuche ist kurz, gut lesbar und nützlich, wenn etwas schiefgegangen ist.

- Protokollieren Sie Start und Ende wichtiger Arbeitsabläufe.
- Protokollieren Sie fehlgeschlagene Validierungen und fehlgeschlagene Deployment-Reviews.
- Protokollieren Sie Exceptions serverseitig mit genügend Details für einen Entwickler.
- Protokollieren Sie keine Passwörter, License Keys oder privaten Benutzerdaten.
- Fügen Sie Zeitstempel hinzu, damit die Reihenfolge der Ereignisse klar ist.

3. Deployment-Ziele

Ziel	Prüfungen auf Junior-Niveau
IIS	App-Ordner veröffentlichen, <code>Web.config</code> prüfen, bei Bedarf das Hosting Bundle kontrollieren, den App-Pool bestätigen, die URL testen.
Kestrel / Self-Hosting	Umgebungsvariablen, Ports, Reverse Proxy, HTTPS und JSON-Konfigurationsdateien prüfen.
Cloud	App-Einstellungen, Secrets, statische Dateien, Logging und die Einrichtung von Deployment-Slot bzw. Staging prüfen.

4. Verbindliche Release-Checkliste

- `Web.config` geprüft und Debug-Modus korrekt gesetzt.
- Wisej.NET License Key sicher verwaltet.
- Einstellungen für Standard-/Startfenster, Theme und URL geprüft.
- Themes und statische Ressourcen enthalten.
- Log-Ziel konfiguriert.
- Konzept für Authentifizierung und Autorisierung geprüft.
- Sensible Dateien sind nicht öffentlich herunterladbar.
- Release-Build vor dem Deployment lokal getestet.
- Anforderungen des Deployment-Ziels geprüft.



Sie sind bereit für das Lab

Als Nächstes: Sehen Sie sich den Video-Walkthrough zum Deployment-Review an, bestehen Sie den Wissenstest und erstellen Sie dann eine Seite mit einer Release-Checkliste – mit Pflicht- und optionalen Prüfungen, Deployment-Notizen, einem Log zur Fehlersuche und einer Paketstatus-Meldung, die sich ändert, sobald alle Pflichtprüfungen erledigt sind.