den Alternativtext im Dialog Objektexportoptionen eintragen (Objekt → Objektexportoptionen). Der Alternativtext wird auch für den Export einer barrierefreien PDF-Datei benötigt.

Die gerade genannten Anforderungen für Bilder können mit dem Skript in Unterkapitel 13.7 geprüft werden.

Dateinamen

Leer- und Sonderzeichen in den Dateinamen von verknüpften Dateien können zu Problemen bei älteren Lesegeräten führen. Um dem vorzubeugen, kann man leicht mit dem Skript *relinkImages.jsx* aus Unterkapitel 13.4.2 die Dateinamen bereinigen.

13.2 Problematische Zeichen prüfen

Durch den Neuumbruch im E-Book können manuelle Trennungen und Umbrüche, die für die Druckausgabe eingefügt wurden, sichtbar werden. Wenn Sie viele E-Books lesen, ist Ihnen dieses Problem vermutlich schon aufgefallen. Die folgenden Arbeitstechniken führen zu Problemen beim E-Book-Export und sollten bereits beim Dokumentaufbau vermieden werden:

- Trennungen, die durch Bindestriche ausgeführt werden, bleiben im Export erhalten. Verwenden Sie ausschließlich *bedingte Trennstriche*.
- *Harte Zeilenumbrüche* sollten nur dann eingesetzt werden, wenn diese auch im Export enthalten sein sollen.
- *Tabulatoren* und das Zeichen *Einzug hier* können nicht dargestellt werden. Das gilt insbesondere für den Aufbau von *Listen*.
- *Leerzeilen* oder *Leerzeichen* sollten nicht zur Formatierung eingesetzt werden.

Eine automatische Korrektur der oben genannten Punkte ist nicht möglich, da insbesondere Trennungen durch Bindestriche nicht eindeutig von Koppelwörtern bzw. Zusammenschreibungen unterschieden werden können. Aber auch die anderen Zeichen bedürfen normalerweise einer manuellen Korrektur. Per Skript lassen sich aber die problematischen Stellen im Dokument auffinden und dann bei Bedarf korrigieren.

Interaktive Korrektur

Das Skript *findCriticalText.jsx* erledigt diesen Job. Es springt zu den entsprechenden Textstellen und bietet die Möglichkeit, für jeden Einzelfall zu entscheiden. Der Fokus liegt auf der interaktiven Bearbeitung der gefundenen Fehlerstellen, das Skript beinhaltet zusätzlich die Möglichkeit, Suchen/Ersetzen-Funktionen generell anzuwenden. Zur Sicherheit wird das Dokument vor der Korrekturphase gespeichert. Zum Testen können Sie das Dokument *findCriticalText.idml* verwenden. Das Skript ist auch für die Konvertierung in andere digitale Ausgabeformate hilfreich.

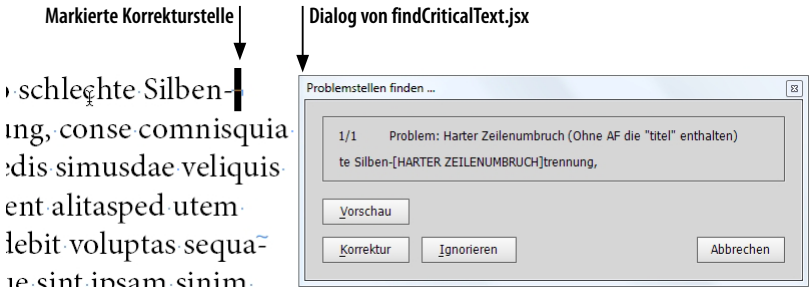

Abb. 108
Korrekturfenster von *findCriticalText.jsx*

Korrekturen durchführen

Wenn Sie den Button KORREKTUR wählen, wird die Korrektur ausgeführt, mit IGNORIEREN wird die Fehlerstelle nicht korrigiert. In beiden Fällen springt das Skript danach automatisch zur nächsten Fehlerstelle. Wenn Sie VORSCHAU anklicken, wird die Korrektur im Dokument ausgeführt, die Auswahl bleibt bei der aktuellen Fehlerstelle. Wenn Sie danach IGNORIEREN wählen, wird die Korrektur rückgängig gemacht, wenn Sie KORREKTUR anklicken, bleibt die Korrektur bestehen.

Veränderungen vornehmen

Der Dialog ist nicht modal, so dass Sie auch direkt im Dokument eingreifen können. Sie dürfen allerdings nicht in den Textbereich vor der Fehlerstelle eingreifen, da sich dann ggf. der Index der Fehlerstelle verschiebt und die Korrekturfunktion bei Folgefehlern nicht mehr zuverlässig arbeitet. Format- oder Textänderungen, die keinen Einfluss auf den Index haben, sind möglich. Wenn Sie nach der VORSCHAU Änderungen am Dokument vornehmen, kann die durch die Vorschau ausgelöste Korrektur nicht mehr rückgängig gemacht werden.

Auswahl oder Dokument

Das Skript bearbeitet entweder den gerade ausgewählten Textbereich oder das komplette Dokument. Die Entscheidung wird davon abhängig gemacht, ob beim Start des Skripts ein Textbereich ausgewählt ist.

Das vorgefertigte Skript soll als Grundlage für eigene Anpassungen dienen. Es enthält die folgenden interaktiven Funktionen:

- Tabulatorzeichen können durch einen Leerraum ersetzt werden.
- Trennstriche mit folgendem Umbruch können entfernt werden.
- Mit Bindestrich getrennte Wörter können kontrolliert werden.
- Harte Zeilenumbrüche, die in Absätzen stehen, deren Absatzformat nicht `Titel` enthält, können zu einem Leerraum korrigiert werden.

Außerdem werden immer die folgenden Korrekturen ausgeführt:

- Leerraum zu Beginn eines Absatzes wird entfernt.
- Durch die Formatoption BUCHSTABENART → GROSSBUCHSTABEN erstellte Großbuchstaben werden zu echten Großbuchstaben konvertiert.

Das Skript erweitern

Bevor man das Skript produktiv einsetzt, sollten die Anforderungen für das konkrete Projekt geklärt sein. Beispielsweise ist die Voraussetzung, dass die Namen aller Überschriften-Absatzformate den String `titel` enthalten, in der Praxis nicht immer gegeben.

Im Folgenden wird nicht das ganze Skript, sondern nur die Anpassung der Suchen/Ersetzen-Funktion und das Einbinden eigener Funktionen zur Korrektur beschrieben. Die Konfiguration des Skripts wird mit dem Objekt `fcList` ganz zu Beginn des Skripts vorgenommen.

Listing 135 Das Konfigurationsobjekt

```
var fcList = {
  iaGREP : [
    {findGREP:"-[\\n\\r]", changeString:"",
     errorText:"Trennung + Umbruch?",
     solutionText:"Trennstrich und Umbruch werden entfernt"},
    {findGREP:"(?<=[\\l\\u\\d])-(?=[\\l\\d])", changeString:"",
     errorText:"Unklare Trennung?",
     solutionText:"Trennstrich wird entfernt"}
  ],
  defaultGREP : [
    {findGREP:"^\\h+", changeGREP:""},
  ],
  iaFunctions : [
    {findFunction:"harterZeilenumbruchAusserTitel",
     changeFunction:"fix_harterZeilenumbruchAusserTitel",
     errorText:"Harter Zeilenumbruch (Ohne AF die \"titel\"
     enthalten)",
     solutionText:"Zu Leerraum ersetzen"},
  ],
  defaultFunctions : [
    {findChangeFunction:"echteGrossbuchstabenErstellen"},
  ]
}
```

Funktion des Skripts

Im Konfigurationsobjekt `fcList` können GREP-Suchen sowie selber erstellte Funktionen eingetragen werden. Grundsätzlich wird unterschieden zwischen Anweisungen, die vom Benutzer interaktiv gelöst werden müssen (`ia...`), und Anweisungen, die immer ausgeführt werden (`default...`). Die Eigenschaften `iaGREP` und `defaultGREP` nehmen Suchen/Ersetzen-Anweisungen auf. In `iaFunctions` können Suchen/Ersetzen-Funktionen eingetragen werden. In `defaultFunctions` werden Korrekturfunktionen eingetragen, die ein Problem selbstständig lösen. Die vier Eigenschaften des Objekts `fcList` enthalten jeweils einen Array. Informationen zur Erstellung von eigenen Objekten finden Sie auf Seite 121.

Im Beispiel-Code werden in `iaGREP` verschiedene Arten von fehlerhaften Trennungen lokalisiert. In `iaFunctions` werden harte Zeilenumbrüche außerhalb von Überschriften über eine eigene Korrektur-

funktion lokalisiert. Automatisch werden in `defaultGREP` fehlerhafte Leerräume gelöscht und in `defaultFunctions` falsch formatierter Text in echte Großbuchstaben konvertiert.

Interaktive Suchen/ Ersetzen-Anweisungen

Der Array in der Eigenschaft `iaGREP` enthält Elemente, die jeweils eine Suchen/Ersetzen-Anweisung und deren Beschreibung beinhalten. Die Eigenschaft `findGREP` muss eine GREP-Suchanfrage für den zu suchenden Fehler enthalten. Im Beispielskript werden zwei Probleme gesucht. Mit `-[\n\r]` wird nach einem Bindestrich, gefolgt von einem harten Zeilenumbruch oder Absatzende gesucht. Möglicherweise fehlerhafte Trennungen werden mit `(?<=[\l\u\d])-(?=[\l\d])` lokalisiert. Es werden Bindestriche zwischen einem beliebigen Buchstaben oder einer Zahl und einem Kleinbuchstaben oder einer Zahl gefunden.

Für die GREP-Suchen in diesem Skript ist der Einsatz von Look Around Assertions (Unterkapitel 10.6) wichtig, weil die Fehlerstelle nur die eigentlich fehlerhaften Zeichen enthalten darf. Der GREP muss als String übergeben werden, entsprechend muss der Backslash mit `\\` doppelt maskiert werden.

Die Eigenschaft `changeString` enthält einen String für das Korrekturzeichen. Wenn die gefundene Fehlerstelle gelöscht werden soll, übergibt man einen leeren String. Sie dürfen hier keinen GREP verwenden, weil das Skript nur mit festen Werten korrigieren kann. Der Text der Eigenschaft `errorText` wird als Beschreibung im Dialog angezeigt, der Text der Eigenschaft `solutionText` wird als Hinweis beim Button KORREKTUR angezeigt. Sie können hier weitere Objekte mit eigenen Suchanfragen hinzufügen, vergessen Sie dabei nicht das Komma zwischen den Objekten.

Default Suchen/ Ersetzen-Anweisungen

In der Eigenschaft `defaultGREP` ist ein Array mit Objekten gespeichert, die eine Suchen/Ersetzen-Anweisung per GREP steuern. Die Objekte sind ähnlich zu den Objekten in `iaGREP` aufgebaut. Sie enthalten die Eigenschaft `findGREP` für die Suche und `changeGREP` für die Ersetzung. In `changeGREP` kann, wie der Name vermuten lässt, auch eine GREP-Ersetzung mit Zugriff auf die Fundstelle eingetragen werden. Im Standardskript werden mit der Suche nach `^\h+` alle Leerräume zu Beginn eines Absatzes entfernt. Diese Leerräume sind immer Fehler und können ohne Benutzerinteraktion gelöscht werden. Falls Sie InDesign-Spezialzeichen wie Indexmarken verwenden, beachten Sie bitte auch das Unterkapitel 10.16 zu diesem Thema.

Interaktive Suchen/ Ersetzen-Funktionen

Die Eigenschaft `iaFunctions` ist recht ähnlich aufgebaut. Zur Suche und Korrektur werden aber in den Eigenschaften `findFunction` und `changeFunction` die Namen von selbst erstellten Funktionen übergeben. Das hat den Vorteil, dass innerhalb dieser Funktionen alle InDesign-Funktionen angesteuert werden können und mit Hilfe von

if-Abfragen die Steuerung verfeinert werden kann. Im Beispielskript wird die Funktion `harterZeilenumbruchAusserTitel` für die Suche und `fix_harterZeilenumbruchAusserTitel` eingesetzt. Die beiden Funktionen werden hier exemplarisch vorgestellt; wenn Sie eigene Funktionen hinzufügen wollen, müssen Sie diese nach dem gleichen Schema erstellen.

Listing 136 `harterZeilenumbruchAusserTitel()`

```
function harterZeilenumbruchAusserTitel (context) {
  if (app.findChangeGrepOptions.hasOwnProperty ("searchBackwards")){
    app.findChangeGrepOptions.searchBackwards = false;
  }
  app.findGrepPreferences = NothingEnum.NOTHING;
  app.changeGrepPreferences = NothingEnum.NOTHING;
  app.findGrepPreferences.findWhat = "\\n";
  var results = context.findGrep(true);
  for (var i = results.length - 1; i >= 0; i--) {
    result = results[i];
    if (result.appliedParagraphStyle.name.toLowerCase().
    indexOf("titel") > -1) {
      results.splice (i, 1);
    }
  }
  app.findGrepPreferences = NothingEnum.NOTHING;
  app.changeGrepPreferences = NothingEnum.NOTHING;
  return results;
}
```

Aufbau der Suchfunktion

Die Funktion für die Suche übernimmt einen Parameter, im Beispiel `context`, der den Suchbereich enthält – das kann ein Objekt vom Typ `Document` oder `Text` sein. Für diese Funktion spielt das keine Rolle, weil beide Objekte die Methode `findGrep()` enthalten. Wenn Sie eigene Funktionen entwickeln und auf Eigenschaften oder Methoden zugreifen, die nicht in beiden Objekten enthalten sind, müssen Sie den Typ des Objekts vorab bestimmen und darauf reagieren (→ Seite 162). Die Funktion muss einen Array mit Textstellen (den Fehlern) zurückliefern – im Beispiel sind die Fehler in `results` enthalten. Dieser Array wird dann im Dialog abgearbeitet.

Die Funktion enthält die Suche nach einem GREP, deren Ergebnis in einer for-Schleife geprüft und verfeinert wird. Die Idee ist, dass alle harten Zeilenumbrüche, die innerhalb von Überschriften stehen, für den EPUB-Export erhalten bleiben sollen und deswegen auch nicht als Fehler gemeldet werden müssen. In der Praxis muss die Namensprüfung an die tatsächlichen Gegebenheiten angepasst werden.

Das Vorgehen ist ähnlich dem FindAndDo-Skript aus Unterkapitel 4.9. Innerhalb der for-Schleife wird allerdings keine Ersetzung vorgenommen, sondern es wird geprüft, ob der Name des Absatzformats den String `titel` enthält. Wenn dies der Fall ist, steht der harte Zeilenumbruch in einem Titel und soll nicht entfernt werden. Die Textstelle

muss aus dem Array mit den Fehlern entfernt werden. Dazu wird die Array-Funktion `splice()` eingesetzt, als erster Parameter wird der Index des Elements, als zweiter Parameter die Anzahl der zu entfernenden Elemente übergeben. Nach der Prüfung wird der bereinigte Array mit `return` zurückgeliefert.

Listing 137 `fix_harterZeilenumbruchAusserTitel()`

```
1  function fix_harterZeilenumbruchAusserTitel (args) {
2    var currentError = args[0];
3    currentError.contents = " ";
4  }
```

Aufbau der Ersetzungsfunktionen

Die Funktion erhält als Parameter einen Array, der als einziges Element ein Objekt vom Typ `Text` an der Stelle des Fehlers enthält. Der Zugriff auf das Textelement in der zweiten Zeile der Funktion muss also so oder ähnlich in allen Ersetzungsfunktionen durchgeführt werden. Der Grund für die Parameterübergabe ist der Aufruf der Funktion im Hauptskript.

Die Korrektur an dieser Stelle ist einfach, es wird der Inhalt der Textstelle mit einem Leerzeichen ersetzt. In der Korrekturfunktion kann aber ausgehend von der Textstelle beliebig programmiert werden. Denkbar ist das Zuweisen von Formaten, beispielsweise beim Umbau von Aufzählungen, oder die Korrektur von Abständen, die durch leere Absätze erzeugt wurden.

Default Korrekturfunktionen

In der Eigenschaft `defaultFunctions` werden Objekte gesammelt, die Funktionen beschreiben, die generell ausgeführt werden. Es wird nur der Funktionsname in der Eigenschaft `findChangeFunction` benötigt. Die Funktion muss im Skript enthalten sein und sich um die Korrektur kümmern. Als Parameter wird das Objekt, in dem die Ersetzungen vorgenommen werden sollen, übergeben. Die Funktion `echteGrossbuchstabenErstellen()` aus dem Beispielskript wandelt mit der Funktion Buchstabenart → Grossbuchstaben erstellte Großbuchstaben in echte Großbuchstaben und kann exemplarisch für die Erstellung von Korrekturfunktionen analysiert werden. Eine ähnliche Funktion könnte für die Konvertierung von Buchstabenart → Kapitälchen geschrieben werden.

Technische Details zum Skript

Session

Das Skript arbeitet mit einer ScriptUI-Palette (→ Seite 205). Paletten sind nicht modal, so dass Sie auch im Dokument Änderungen vornehmen können, während die Palette angezeigt wird. Damit die Palette auch nach dem Ende des Skripts noch erreichbar ist, wird zu Beginn des Skripts mit `#targetengine` eine Session angelegt (→ Seite 206).

Globales Objekt

Um später die Funktionen aufzurufen, wird das Objekt `global` benötigt. Es repräsentiert den globalen Bereich und wird normalerweise implizit verwendet; wenn Sie `app` oder `alert()` verwenden, greifen Sie