Iwein – digital

Handbuch zur TEI-Kodierung
und zur Transkription

1. Einleitung

Dieses TEI-Handbuch wurde für das Projekt der digitalen Edition des *Iwein* Hartmanns von Aue erstellt. Es entsteht in Abhängigkeit von den Richtilinien, die HeiEditions über das Validierungsschema und die dazugehörige **KurzDoku** festlegt. Das Handbuch dient vor allem als Leitfaden für die Bearbeiterinnen und Bearbeiter der Handschriften.

Die XML-Dateien für die TEI-Codierung der Transkriptionen sind als ***templates*** oder Vorlagen bereits vorgefertigt. In diesen Vorlagen ist die Struktur der Verspaare bereits angelegt und 8200 Verse sind bereits nummeriert. Jeder Vers enthält sechs leere <w>-Elemente, die bei der Transkription nur ausgefüllt werden müssen. Es gibt sogar *templates* mit bereits kodiertem Reimpunkt am Versende oder mit vorgefertigten Tags für Versalien oder Buchstabenrubrizierung am Versanfang, die je nach Handschrift herangezogen werden können. Auf diese Weise werden häufige mechanische Aktionen erspart und die Transkription erleichtert.

Wir arbeiten mit dem **Oxygen** XML-Editor und installieren dazu das von Gustavo Fernández Riva (UB Heidelberg) entwickelte ***framework*** "IWDDocument". Dieses *framework* ist für die Arbeit im Author-Modus konzipiert, in dem – im Unterschied zum Text-Modus – die XML-Elemente und Attribute nicht sichtbar sind. Das *framework* erlaubt es mit einfacher Menü-Auswahl oder mit Tastatur-shortcuts oft benötigte Codierungen fehlerfrei und schemakonform einzugeben. Mit der Tabulator-Taste kann man einfach von einem Wort zum nächsten springen.

Die Angaben in diesem Handbuch beschreiben die TEI-Codierung, die im Textmodus sichtbar ist. Am Ende jedes Abschnitts werden aber in roter Farbe und am rechten Rand auch die Eingabemodi für das *framework* im Author-Modus gegeben; fehlen die Angaben, dann muss die beschriebene Kodierung im Text-Modus durchgeführt werden.

Gearbeitet wird mit dem Zeichensatz **Junicode**, der für mittelalterliche Texte sehr geeignet ist.

Inhalt

1. XML-Deklaration und TEI-Root-Element

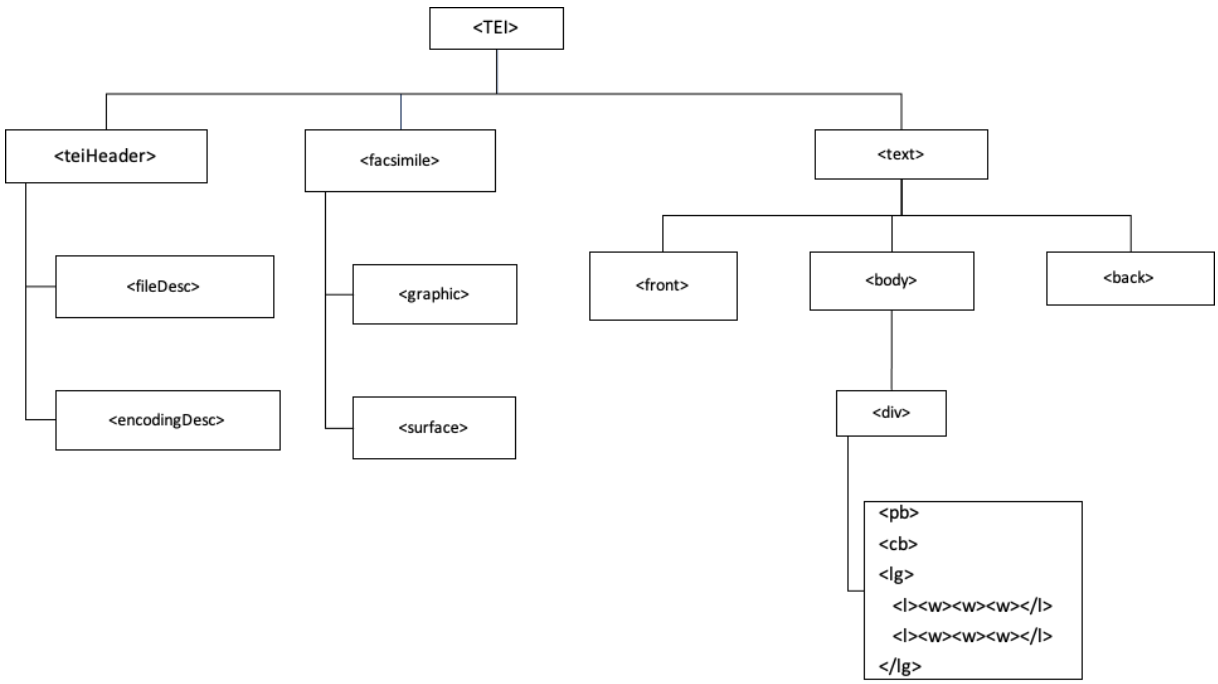
a) Die Deklarationen für XML und Schema sind für alle Dateien folgende:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-model href="https://digi.ub.uni-heidelberg.de/schema/tei/heiEDITIONS/tei_hes.rng"
    type="application/xml" schematypens="http://relaxng.org/ns/structure/1.0"?>
<?xml-model href="https://digi.ub.uni-heidelberg.de/schema/tei/heiEDITIONS/tei_hes.rng"
    type="application/xml" schematypens="http://purl.oclc.org/dsdl/schematron"?>
```

b) Das Root-Element <TEI> enthält folgende Attribute und Werte:

```
<TEI xmlns="http://www.tei-c.org/ns/1.0" xmlns:hei="https://digi.ub.uni-heidelberg.de/schema/tei/
    heiEDITIONS" ana="hc:IWDDocument">
```

c) Die Struktur der Datei sieht dann wie folgt aus:



2. Der TEI-Header

Das Element <teiHeader> wird zunächst nur mit zwei Kindelementen gefüllt: <fileDesc> und <encodingDesc>.

a) Die Datei-Beschreibung <fileDesc> enthält ein <titleStmt>, ein <publicationStmt> und eine <sourceDesc>.

- Das <titleStmt> enthält das Element <title> mit dem Inhalt: "Iwein-Hs. X"; das Element <author> mit dem Inhalt "Hartmann von Aue"; und das Element <respStmt> mit den Kindelementen <resp> (Inhalt: "transcription") und <persName> (Inhalt: "Name des Transkribenden").
- Das <publicationStmt> enthält lediglich das Kindelement <publisher> mit dem Inhalt "UB Heidelberg".

– Die <sourceDesc> enthält das Element <listWit> mit dem Kindelement <witness>; dieses Element erhält als Attribut eine @xml:id mit dem Wert der Sigle der Handschrift. <witness> hat als Kindelement <msDesc> und dieses wiederum als Kindelement <msIdentifier>. Kindelemente des letzteren sind <settlement> (Inhalt: Bibliotheksort), <repository> (Inhalt: Bibliotheksname) und <idno> (Inhalt: Signatur der Handschrift).

b) Die Codierungs-Beschreibung muss folgende sein:

<encodingDesc><listPrefixDef><prefixDef ident="hc" matchPattern="(.+)" replacementPattern="https://lod.ub.uni-heidelberg.de/ontologies/heieditions/hc/current/$1"/></listPrefixDef> </encodingDesc>

3. Das Element <facsimile>

Zwischen dem TEI-Header und dem Text steht das Element <facsimile>. Es wird erst am Ende mit der Information zu Seiten (<surface>), Abbildungen (<graphic/>) und Bildbereichen (<zone/>) gefüllt.

Jede Seite der Handschrift wird als <surface> erfasst. Das Element hat die Attribute @ana (mit dem Wert "hc:Page"), @xml:id (dessen Wert die Sigle der Handschrift und die Seitenzahl ist, z.B. "b_12v") und @n (mit dem Wert der Seitenzahl, z.B. "12v").

Als erstes Kindelement der <surface> steht das Element <graphic/> mit dem Attribut @url und als Attributwert die url der digitalen Abbildung dieser Seite. Das Element ist leer.

Als weitere Kindelemente von <surface> folgen nun so viele Elemente <zone/> wie die Seite Bildbereiche hat. Bildbereiche der Seite sind Spalten und evtl. existierende Randbereiche. Eine einspaltig beschriebene Seite hat also ein einziges Element <zone>, eine zweispaltig beschriebene Seite mit einer Lagenzählung im unteren Rand hat drei <zone>-Elemente. <zone>-Elemente für *Spalten* haben die Attribute @ana (mit dem Wert "hc:Column"), @xml:id (dessen Wert ist "Sigle der Handschrift"_"Seitenzahl"-"Spalte", z.B. "b_12v-a") und @n (mit dem Wert des Spaltenbuchstabens, z.B. "a"). <zone>-Elemente für *Randbereiche* haben die Attribute @ana (mit dem Wert "hc: MarginalZone"), @xml:id (dessen Wert ist "Sigle der Handschrift"_"Seitenzahl"-"ein Buchstabe mehr als die letzte Spalte", z.B. "b_12v-b") und @hei:placeRef (in der Regel mit dem Wert "hc:PageBottom", ggf. ergänzt durch "hc:PageMarginRight"). Das Element <zone/> ist leer.

Beispiel:

```
<surface ana="hc:Page" xml:id="b_12v" n="12v">
   <graphic url="https://digi.ub.uni-heidelberg.de/diglit/cpg391/0028/image"/>
   <zone ana="hc:Column" xml:id="b_12v-a" n="a"/>
   <zone ana="hc:MarginalZone" xml:id="b_12v-b" hei:placeRef="hc:PageBottom
            hc:PageMarginRight"/>
</surface>
```

Zu Randbereichen wie Lagensignaturen und Reklamanten => 27; zu xml:id => 25.

Framework: Actions → Facsimile

Nicht geordnete Handschriften wie A sollten allerdings manuell überprüft werden. Randzonen wie Lagensignaturen oder Randanmerkungen müssen ebenfalls manuell eingetragen werden.

4. Das Element <text>

Auf das Element <facsimile> folgt nun der Text. Das Element <text> enthält das Attribut @ana, dessen Wert aus zwei Bestandteilen besteht: zum einen "hc:CourtlyRomance" und zum anderen, je nach Fall, "hc:CompleteExpression" oder "hc:ExpressionFragmentThroughLoss".

5. Titel und Kolophon

Das Element <text> enthält immer das Element <body>. Es kann auch die Elemente <front> und/oder <back> enthalten. In <front> werden die Titel notiert, die manche Schreiber dem Werk voranstellen und die nicht zu Hartmanns Text gehören. Falls so ein Titel in der Handschrift existiert, wird ein <front>–Element angelegt. Als erste Kindelemente stehen <pb> und <cb> (=> 7a und 7b). An dritter Stelle wird ein Element <docTitle> angelegt mit einem Kindelement <titlePart>. In diesem wird nun der Titel eingetragen, tokenisiert (=> 9b) und mit Angabe der <lb/> (=> 7c).

Beispiel:

```
<front>
  <pb n="5v"/><cb n="c"/>
    <docTitle><titlePart>
      <lb n="49"/><hi hei:color="Red"><w>Von</w><w>künig</w><w>Artus</w><w>hochzeit</w>
  </titlePart></docTitle>
</front>
```

In <back> hingegen werden typischerweise Schreiberkolophone eingetragen. Das Element <back> enthält ein Kindelement <trailer> mit dem Attribut @ana und dem Wert "hc:Colophon".

Beispiel:

```
<back>
   <trailer ana="hc:Colophon">
      <w>Amen</w>
   </trailer>
</back>
```

Manchmal enthalten Kolophone Verzierungen, die wir als <metamark> mit dem Attribut @ana und dem Wert "hc:DecorativeMarker" taggen.

6. Der Text

Das Element <body> enthält nur das Kindelement , ich welchem der Text des *Iwein* steht. Der Text wird strukturiert in Verspaare und Verse erfasst.

7. Seite / Spalte / physische Zeile

a) Der Beginn einer neuen Seite wird getaggt mit <pb/>, auch der Beginn der ersten Seite (evtl. bereits in <front>, => 5). Dieses Element ist immer leer. Es steht – wie alle Elemente – immer möglichst weit 'oben'

in der xml-Struktur: Bei Handschriften mit abgesetzten Versen also hinter dem Schluss der letzten <lg> der vorangehenden Seite, oder, wenn von einem Verspaar ein Vers auf jeder Seite steht, hinter dem Schluss der letzten <l>. Bei Handschriften ohne abgesetzten Versen allerdings steht das Element an der ensprechenden Stelle. Diesem Element folgt immer das Element <cb/>. Es enthält immer das Attribut @n; der Wert des Attributs ist die Blatt- und Seitenzahl, z.B.: <pb n="1v"/>, oder <pb n="2r"/>

Framework: Insert Tags → pb

b) Der Beginn einer neuen Spalte wird getaggt mit <cb/>. Dieses Element ist immer leer. In einspaltigen Handschriften steht es immer hinter dem Element <pb/>. Bei mehrspaltigen Handschriften steht es außerdem hinter dem letzten Vers oder Verspaar der vorangehenden Spalte und vor der ersten Zeile der neuen Spalte. Es enthält immer das Attribut @n; der Wert des Attributs ist der Buchstabe für die jeweilige Spalte (a, b, c), z.B.: <cb n="a"/>, oder <cb n="b"/>

Framework: Insert Tags → cb

c) Der Beginn einer neuen physischen Zeile wird getaggt mit <lb/>. Dieses Element ist immer leer. In Handschriften mit abgesetzten Versen steht es immer als erstes Kind-Element von <l>. Es enthält enthält immer das Attribut @n; der Wert des Attributs ist die Nummer der physischen Zeile in der aktuellen Spalte, z.B. <l><lb n="23"/><w>...

Framework: Insert Tags → lb; *Shortcut:* ctrl/cmd + l

Zu Zeilenüberhängen => 17.

8. Verspaar und Vers

a) Jedes Verspaar steht innerhalb des Elements <lg>. In der Regel benötigt dieses Element kein Attribut, doch es besteht die Möglichkeit, bei Drei-Reimen oder bei durch Reimänderungen eventuell entstehenden Strophen dies mit dem Attribut @ana zu markieren. Der Wert des Attributs wäre dann "hc:Tercet" oder "hc:Quatrain".

Framework: Insert Tags → new stanza

b) Jeder Vers steht innerhalb des Elements <l>. Dieses Element führt das Attribut @n; der Wert des Attributs ist die Versnummer nach Lachmann.

Bei Versumstellungen folgt die Transkription der physischen Versanordnung der Handschrift. Das bedeutet, dass die Nummern der <lb>-Elemente stets konsekutiv bleiben; die Nummern der <l>-Elemente hingegen werden angepasst. Zum Beispiel:

<l n="113"><lb n="24"/>
<l n="112"><lb n="25"/>

Fehlt ein Vers oder eine Gruppe von Versen, stehen sie schlicht nicht in der Transkription:

<l n="101"><lb n="24"/>
<l n="112"><lb n="25"/>

Bei hinzugefügten Versen werden die notwendigen <l> und <lg>-Elemente hinzugefügt. Die Numerierung der <l>-Elemente im @n-Attribut ist die des letzten mit Lachmann übereinstimmenden Verses, gefolgt von einem Komma und einer zweiten konsekutiven Zahl, z.B.:

<l n="117">

<l n="117,1">
<l n="117,2">

Framework: Attributes → verse_number; *Shortcut:* shift + ctrl/cmd + V
Attributes → line_number; *Shortcut:* shift + ctrl/cmd + L
Insert Tags → new verse

9. Wort und Tokenisierung

a) Jedes Wort wird getaggt mit dem Element <w>. Dies nennt man auch ein *token*.

Framework: Insert Tags → new word; *Shortcut:* shift + Enter

b) Tokenisierung

Zusammen- und Getrenntschreibung ist in Handschriften bei weitem nicht immer eindeutig. Im Zweifelsfall werden die Worte so erfasst, wie es die Wörterbücher vorsehen: stehen zwei Worte nahe beieinander, sind aber nicht eindeutig zusammengeschrieben, werden sie als zwei Worte erfasst; sind innerhalb eines Wortes zwei Buchstaben ein klein wenig voneinander getrennt, ohne dass es sich aber um eine eindeutige Getrenntschreibung handelt, wird das Wort zusammengeschrieben.

Bei deutlicher Getrenntschreibung innerhalb eines grammatischen Wortes (z.B. bei Verbpräfixen) wird das Wort als solches in einem einzigen <w>-Element erfasst und die Trennung mit einem Spatium markiert: <w>ge lesen</w>. Die Spatien sollen bei der Normalisierung getilgt werden können.

Bei eindeutiger Zusammenschreibung werden beide Worte als getrennte Tokens erfasst und ihre Zusammenschreibung über ein @join-Attribut gekennzeichnet: das erste Token erhält das Attribut @join="right" das zweite @join="left".[1] Sollten drei Worte zusammengeschrieben worden sein, erhält das mittlere das Attribut @join="both".

Framework: Attributes → join; *Shortcut:* ctrl/cmd + J

Besondere Regeln für die Tokenisierung:

- Pronominaladverbien und trennbare Verbpräfixe werden immer getrennt. Beispiele:
  daran → <w join="right">dar<w/><w join="left">an</w>
  warumbe → <w join="right">war<w/><w join="left">umbe</w>
  abebrechen → <w join="right">abe<w/><w join="left">brechen</w>
- Bei Kontraktionen (Proklise, Enklise und Krasis) werden die Worte ebenfalls getrennt:
  - Gehört der Vokal im Bereich der Krasis eindeutig (auch) zu dem zweiten Wort, so wird vor ihm getrennt; gehört er nur zu dem ersten Wort, wird nach ihm getrennt. Beispiele:
    sageter → <w join="right">saget<w/><w join="left">er</w> (< *sagete er*)
    zen → <w join="right">z<w/><w join="left">en</w> (< *ze den*)
    zun → <w join="right">zu<w/><w join="left">n</w> (< *zuo den*)
  - Gehört ein Konsonant im Bereich der Krasis eindeutig (auch) zu dem ersten Wort, so wird nach ihm getrennt; gehört er nur zur zweiten Form, so wird vor ihm getrennt. Beispiele:
    mirz → <w join="right">mir<w/><w join="left">z</w> (< *mir ez*)
    alsi → <w join="right">als<w/><w join="left">i</w> (< *alse si*).
  - dreifache Kontraktion:

[1] Das "left"-Attribut ist nicht notwendig. Wir fügen es trotzdem hinzu, das *framework* tut dies automatisch.

destwar → <w join="right">de</w><w join="both">st</w><w join="left">war</w>
(ebenso auch für *deiswar* oder *deswar*)

– Bei Kontraktion mit dem Pronomen der 2. Pers. Sing *du* stellen wir das *t* oder *d* an der Grenze immer zum Personalpronomen.[2]
mahtu → <w join="right">maht<w/><w join="left">u</w>
mustu → <w join="right">muost<w/><w join="left">u</w>
soltu → <w join="right">solt<w/><w join="left">u</w>
bistu → <w join="right">bist<w/><w join="left">u</w>
magstu → <w join="right">mags<w/><w join="left">tu</w>
sagestu → <w join="right">sages<w/><w join="left">tu</w>
wellestu → <w join="right">welles<w/><w join="left">tu</w>

10. Grapheme und Majuskeln

a) Obwohl alle Zeichen so exakt wir möglich transkribiert werden sollen, ist eine Transkription nicht die exakte Reproduktion der Handschrift. Wir übertragen mit der Hand geschriebene Texte in eine mechanische Schrift. Das bedeutet, dass ein variables und polymorphes zu einem einförmigen Zeichensystem vereinfacht und vereinheitlicht wird. Wir reproduzieren also nicht die verschiedenen Allographen der Handschriften (unterschiedliche Formen von z, d, r, s usw.), sondern schreiben immer nur die entsprechenden Grapheme neuzeitlicher Druckschrift. Einzige Ausnahme ist das Schaft-s "f", weil dies oft schon von mittelalterlichen Schreiben mit "f" verwechselt wurde. Es kann bei der Textprozessierung leicht zu "s" normalisiert werden.

Framework: Special Characters → S; *Shortcut:* ctrl/cmd + S

b) Majuskeln von Minuskeln zu unterscheiden, ist vor allem in späteren Handschriften nicht leicht. Eine Majuskel sollte einen anderen Schriftzug haben als die entsprechende Minuskel (wie heute in aA bB dD eE fF gG). Aber auch so ist es manchmal schwierig, weil Grapheme, die wir als Majuskeln transkribieren würden, plötzlich im Wortinneren erscheinen. Da wir die mittelalterlichen Grapheme nicht reproduzieren, transkribieren wir sinnweise: nur klare Majuskeln am Wortanfang werden als solche erfasst; alles andere sind Minuskeln. Ausgenommen sind alle *litterae notabiliores*, also Initialen, rubrizierte Versanfänge und sonstige hervorgehobene Anfangsbuchstaben, die wir immer als Majuskeln transkribieren.

11. Sonderzeichen, Diakritika und Superskripte

Sonderzeichen werden möglichst genau wiedergegeben. Der Junicode-Zeichensatz bietet in der Regel ausreichende Optionen. Das gilt auch für Diakritika und Superskripte. Allerdings stehen diese über oder unter dem Laut, auf den sie sich funktional beziehen. Etwaige horizontale Verschiebungen, wie sie in den jüngeren Papierhandschriften häufig sind, werden also nicht abgebildet. Ebenfalls wird bei Diakritika nicht jeder Biegungsgrad einer Tilde oder Hakens, wie sie in Papierhandschriften häufig sind, nachgemacht. Alle Superskripte und Diakritika werden vorzugsweise mit kombinierenden Zeichen wiedergegeben; bei

[2] Wir vereinfachen damit die Heidelberger Richtlinien, die hier nach Verbklassen unterscheiden.

der Transkription selbst ist es einfacher, erst mit normalen Tastenkombinationen z.B. ÿ zu schreiben, um erst am Ende das integrierte Trema durch ein kombiniertes Zeichen zu ersetzen.

Framework: Special Characters

12. Initialen

Initialen werden getaggt mit dem Element <hei:initial>. Sie führen drei Attribute: 1) @ana, das den Initialentypus bezeichnet; dessen häufigster Wert ist "hc:Lombard", doch das Schema sieht unter anderem auch vor "hc:FlourishInitial" oder "hc:ScrollworkInitial"; 2) @hei:color, das die Farbe bestimmt, z.B. "Red", "Blue", "Gold"; und 3) @hei:heightLines, das die Höhe der Initiale in Zeilen angibt, dessen Wert also eine volle Zahl ist.

Framework: Insert Tags → Initial; *Shortcut:* ctrl/cmd + I

Dateneingabe im Fenster

13. Hervorhebungen

1) Hervorhebung durch farbige Schrift wird getaggt mit dem Element <hi> mit dem Attribut @hei:color und den Werten "Red", "Green", "Blue" etc. Ist ein ganzes Wort farbig geschrieben, steht das Element <hi> außerhalb des <w>-Elements, ebenso bei mehreren Wörtern. Sind z.B. zwei Zeilen farbig geschrieben, wird das Element vor dem <lb> geschlossen und danach ein neues <hi>-Element geöffnet: <hi hei:color ="Red"><w/><w/><w/></hi><lb n="2"><hi hei:color="Red"><w/><w/></hi>

Framework: Insert Tags → Rubric; *Shortcut:* ctrl/cmd + R

2) Hervorhebung der Versanfänge durch rote Striche (typisch für Papierhandschriften) wird ebenfalls getaggt mit <hi>; diesmal kommt das Attribut @rendition ins Spiel, mit dem Wert "hc:RedStroke", in manchen Fällen auch "hc:RedRetrace". Hervorgehobene Buchstaben am Versanfang werden als Majuskeln transkribiert.

Framework: Insert Tags → Decoration; *Shortcut:* ctrl/cmd + D

3) Hervorhebung der Versanfänge durch Herausrücken der Anfangsbuchstaben in die Versalienspalte wird ebenfalls getaggt mit <hi> und dem Attribut @rendition, doch in diesen Fällen lautet der Attributwert "hc:Versal".

Framework: Insert Tags → Versal; *Shortcut:* ctrl/cmd + W

14. Zeichensetzung und Metazeichen

a) Als Satzzeichen taggen wir (mit <pc>) Zeichen, die in irgendeiner Form eine sprachliche oder syntaktische Bedeutung haben. Das sind in der Regeln Virgeln "/" oder vor allem Reimpunkte "·" (Punkt auf Mitte).

Framework: Insert Tags → pc; *Shortcut:* ctrl/cmd + .

b) Metazeichen sind weder Buchstaben noch Satzzeichen. Das können Worttrennungszeichen, Korrekturzeichen, Einweisungszeichen oder Ähnliches sein. Sie werden getaggt als <metamark> und führen das Attribut @ana, dessen Wert, je nach Funktion des Zeichens, eine der folgenden Optionen sein wird: "hc:Hyphen" (Trennstrich), "hc:RunOverSign" (Zeilenanschlusszeichen), "hc:WordDivider" (Wort-trenner), "hc:CorrectionMark" (Korrekturzeichen), "hc:InsertionMark" (Einweisungszeichen), "hc: TranspositionMark" (Umstellungszeichen), "hc:ReferenceMark" (Verweiszeichen), "hc:TokenMarker"

(Token-Marker)[3], "hc:DecorativeMarker" (Ziermarker), "hc:LineDelimiter" (Zeilenabgrenzungszeichen), "hc:CueInitial" (Anweisung für Initiale), "hc:CueNumeral" (Anweisung für Nummer), "hc:CueParagraphSign" (Anweisung für Paragraphenzeichen).

Die Anweisungen für Initialen ("hc:CueInitial") setzen wir immer unmittelbar vor das Wort, das die Initiale enthält, auf die das Zeichen verweist. Wir versehen sie mit einem Attribut @hei:PlaceRef, dessen Wert meist "hc:InSpace" sein wird, der aber auch ein anderer sein kann (vgl. 8.b).

Función: Insert Tags → metamark; *Dateneingabe im Fenster*

15. Hinzufügungen, Streichungen, Substitutionen

a) Das Element <del> taggt eine Streichung. Die Art der Streichung wird durch das Attribut @rendition angegeben, dessen Wert je nach Fall sein kann: "hc:Erased" (radiert), "hc:Strikethrough" (durchgestrichen), "hc:RedStrikethrough" (rot durchgestrichen), "hc:Overwritten" (überschrieben), "hc:Overpainted" (übermalt), "hc:Underlined" (unterstrichen), "hc:Underdotted" (unterpungiert), "hc:Overdotted" (überpungiert), "hc:Adapted" (angepasst), oder "hc:ImplicitlyDeleted" (implizit getilgt). Die Attribute sind kombinierbar.

Framework: Insert Tags → del; *Dateneingabe im Fenster*

b) Hinzufügungen werden getaggt als <add>. Das Attribut @hei:placeRef gibt an, wo diese Hinzufügung steht: "hc:AboveLine", "hc:BelowLine", "hc:Inline", "hc:PageMarginLeft", "hc:PageMarginRight", "hc:ColumnMarginLeft", "hc:ColumnMarginRight". Es gibt weitere Werte (=> KurzDoku 5.14).

Framework: Insert Tags → add; *Dateneingabe im Fenster*

c) Eine Substitution liegt dann vor, wenn ein Schreiber etwas Geschriebenes streicht und durch etwas anderes ersetzt. Ein Element <subst> enthält zwangsmäßig und ausschließlich die Elemente <del> und <add>. Die Attribute für <del> und <add> sind die oben angegebenen. Es gibt einen besonderen Fall: wenn ein Buchstabe nicht gestrichen sondern durch Überschreibung angepasst wurde, führen beide Elemente das Attribut @rendition mit dem Wert "hc:Adapted".

Framework: Insert Tags → subst; *Dateneingabe im Fenster*

Beispiele:

Betrifft das Phänomen ganze Worte, stehen die Tags *außerhalb* des <w>-Elements:

- Buchstabe im Wort hinzugefügt: <w>so<add hei:placeRef="hc:AboveLine">l</add></w>
- Buchstabe im Wort gestrichen: <w>leit<del rendition="hc:Underlined">e</del></w>
- Buchstabe im Wort ersetzt: <w>w<subst><del rendition="hc:Underdotted hc:Overdotted">o</del><add hei:placeRef="hc:AboveLine">a</add></subst>rt</w>
- Buchstabe im Wort ersetzt: <w>w<subst><del rendition="hc:Adapted">o</del><add rendition="hc:Adapted">a</add></subst>rt</w>
- Wort gestrichen: <del rendition="hc:Strikethrough"><w>überflüssig</w></del>
- Wort hinzugefügt: <add hei:placeRef="hc:ColumnMarginRight"><w>neu</w></add>

[3] Ein Token-Marker ist ein Zeichen, das manchmal bei Worten zum Einsatz kommt, die nur aus einem Vokal bestehen (z.B. *ê*), um anzuzeigen, dass dies kein falscher Wortansatz ist, sondern ein Wort; z.B. |*ê*| oder *ê'*.

- Wort(e) ersetzt: <subst><del rendition="hc:Erased"><w>falsche</w><w>Wörter</w></del><add hei:placeRef="hc:Superimposed"><w>richtige</w><w>Worte</w></add></subst>

16. Zeilensprünge im Versinneren und im Wortinneren

Bei Handschriften, die die Verse nicht absetzen, werden Zeilensprünge im Versinnern normal mit einem <lb>-Element an der entsprechenden Position markiert. Findet der Zeilensprung im Wortinneren statt, entsteht ein sog. *broken word*. In diesen Fällen werden zwei <w>-Elemente gesetzt, je eins für jedes Wortteil. Das erste führt das Attribut @part="I" (initial), das zweite das Attribut @part="F" (final). Dadurch ist eindeutig festgehalten, dass beide <w>-Elemente ein einziges Wort bilden. Das Framwork setzt die Attribute automatisch in beide Elemente ein.

Framework: Attributes → Broken word; *Shortcut:* ctrl/cmd + B

Zwischen beiden Wortteilen steht das <lb/>-Element mit seinem Attribut @n und (sehr wichtig!) auch mit einem Attribut @break, dessern Wert "no" sein muss.

Wird das Wort mit einem Trennungszeichen getrennt, wird dieses mit einem <metamark> getaggt (mit Attribut @ana="hc:Hyphen"), das vor dem <lb/> steht (=> 14)

Beispiele:

<w part="I">min</w><lb n="4" break="no"/><w part="F">ne</w>

<w part="I">vol</w><metamark ana="hc:Hyphen">=</metamark><lb n="4" break="no"/><w part="F">get</w>

Dies ist kompatibel mit Spalten- oder Seitenumbrüchen. Dann stehen *vor* dem <lb/>-Element die entsprechenden Elemente <pb/> und <cb/> (=> 7a/b).

17. Zeilenüberhänge

Wenn Teile des Versendes aus Platzgründen am Ende der vorangehenden oder folgenden Zeile vervollständigt wurden, sprechen wir von Zeilenüberhang. Der Zeilenwechsel wird getaggt mit einem <lb/>-Element, das den Beginn einer neuen Zeile angibt. Dieses Element führt das Attribut @n, dessen Wert die Nummer der Hauptzeile ist, gefolgt von einem *a* (wenn also die Hauptzeile die Zeile 12 ist, so ist die Überhangzeile 12a). Außerdem erhält das <lb/>-Element ein Attribut @ana, das zwei Werte enthalten muss: zum einen "hc:DependentLine" und zum anderen entweder "hc:RunOverLineAbove" (wenn die Überhangzeile oben steht) oder "hc:RunOverLineBelow" (wenn sie unten steht). Oft wird die Überhangzeile durch Striche vom Rest der Zeile getrennt oder mit der 'Mutterzeile' verbunden. Diese Zeichen werden als <metamark> getaggt; sie führen ein Attribut @ana mit dem Wert "hc:RunOverSign".

Framework: Insert Tags → lb Dependent line; *Dateneingabe im Fenster*

Beispiel:

...<w>name</w><lb n="17a" ana="hc:DependentLine hc:RunOverLineAbove"/><metamark ana="hc:RunOverSign">|</metamark><w>treit</w>

Das ist übrigens kompatibel mit Worttrennung:

...<w part="I">vrüme</w><metamark ana="hc:Hyphen">=</metamark><lb n="17a" ana="hc: Dependent Line hc:RunOverLineAbove"/><metamark ana="hc:RunOverSign">|</metamark><w part="F">keit</w>

18. Nicht ausgeführte Zeichen / Zeilen

a) Manchmal wurde Raum für eine Initiale gelassen, diese aber nicht ausgeführt. Dasselbe geschieht selten auch mit Versalien. Wir ergänzen die fehlenden Buchstaben und taggen sie mit dem Element <supplied> als Eltern-element von <hei:initial> oder <hi>. Ergänzte Initialen bekommen kein Farb-Attribut.

Beispiele:

<w><supplied><hei:initial ana="hc:Lombard" hei:heightLines="3">W</hei:initial></supplied>as</w>

<w><supplied><hi rendition="hc:Versal">W</hi></supplied>as</w>

Framework: Insert Tags → supplied

In zumindest einem Fall ließ der Schreiber eine Zeile frei, trug aber nicht den Vers ein. Das haben wir wie folgt getaggt: <l n="1234"><lb n="5"/><supplied><gap/></supplied></l>. D.h., dass wir die Lücke als Lücke *interpretieren*.

19. Verswiederholungen oder sehr ähnliche Verse

Verswiederholungen werden mit einem Attribut im <l>-Element der beiden korrespondieren Verse gekennzeichnet: @hei:Replicates; wenn klar ist, welcher Vers der 'wiederholte' und welcher das 'Original' ist, dann kann das Attribut auch nur im 'wiederholten' Vers stehen. Der Attributwert ist ein Xpath-pointer mit folgender Syntax: "#xpath(//l[@n='23'])", wobei '23' die Nummer des Verses ist, auf den verwiesen wird.

Beispiel:

<l n="23">...</l/>

[...]

<l n="46" @hei:Replicates ="#xpath(//l[@n='23'])">...</l>

Es gibt Fälle, in denen ein Teile einer Zeile den entsprechenden Teile einer anderen Zeile wiederholt (meist weil der Schreiber mitten im Vers in die falsche Zeile gesprungen ist). In diesen Fällen werden die jeweils korrespondieren Versteile mit dem Element <seg> (segment) getaggt und attributiv mit @xml:id und @target aufeinander bezogen.

<l n="6936"><w>Das</w><w>er</w><w>den</w><seg xml:id="p_seg_6936_1"><w>tot</w>
<w>fehen</w><w>fol</w></seg></l>

<l n="6936,1"><w>Das</w><w>doch</w><w>dem</w><w>andern</w><seg hei:replicates=
"#p_seg_6936_1"><w>dot</w><w>fehin</w><w>fol</w></seg></l>

20. Lücken und beschädigter Text

Textverlust durch Beschädigung des Textträgers wird getaggt mit <gap/>. Dieses Element ist immer leer. Wenn der Text nicht ganz verloren, wohl aber deutlich beschädigt ist, taggen wir ihn mit dem Element

<damage>. Beide Elemente führen das Attribut @ana, je nach Fall mit einem der folgenden Werte: hc:Overpainted – hc:Lost – hc:CutOff – hc:TornOff – hc:Stained – hc:Polluted – hc:Perforated – hc:Mould – hc:Faded – hc:ChemicalReagent – hc:Burnt – hc:PastedOver – hc:PeeledOff – hc:RubbedOff – hc:Illegible. Ist im Fall von <damage> ein Lesung möglich, aber unsicher, wird der Text zusätzlich mit dem Kinderlement <unclear> getaggt.

21. Leere Zeilen

In manchen Fällen, bei Handschriften mit abgesetzten Versen, lässt der Schreiber eine Zeile leer und es fehlt gleichzeitig ein Vers (z.B. E 3838, r 1686). Das bedeutet, dass er gemerkt hat, dass hier ein Vers hingehört (weil ihm der Reim oder ein Satzteil gefehlt hat), in seiner Vorlage aber an dieser Stelle keiner vorhanden ist (oder er ist unlesbar geworden). Um diese Erkenntnis des Schreibers zu erfassen kodieren wir solche Fälle wie folgt (die physische Zeile wird mitgezählt, der Vers nicht):

```
<l><lb n="23"/><supplied><gap/></supplied></l>
```

22. Abkürzungen

a) Abkürzungen müssen in der Edition sowohl als Abkürzung wie auch aufgelöst gezeigt werden können. Solche Optionen werden stets in ein <choice>-Element eingebaut. Die <choice>-Konstruktion steht in der Regel innerhalb des <w>-Elements und schließt nur den Buchstaben ein, der die Abkürzung enthält. Alle anderen Buchstaben bleiben außerhalb.

b) Der Buchstabe mit dem Abkürzungszeichen wird getaggt mit dem Element <am>, das nur innerhalb eines <choice>-Gebildes stehen kann. Nicht alle Abkürzungen haben ein Zeichen, z.B. *dz* => daz.

c) Die Auflösung einer Abkürzung steht im Element <ex>. Dieses Element steht direkt im Anschluss an das <am> innerhalb der <choice>. Die Auflösung ist nicht immer mechanisch und folgt grammatischen Regeln, so kann *einē* aufgelöst werden als *einem* oder *einen*; *er*-Abkürzungen müssen in manchen Fällen als *re* aufgelöst werden, usw. Die Markierung und Auflösung von Abkürzungen ist also schon Teil einer editorischen Arbeit und erfordert entsprechendes Ermessen.

Framework: Abbreviations; *Shortcut:* shift + ctrl/cmd + A; *Dateneingabe im Fenster*

Beispiele

un̄ → und

```
<w>u<choice><am>n̄</am><ex>nd</ex></choice></w>
```

v̊ſehē → verſehen

```
<w><choice><am>v̊</am><ex>ver</ex></choice>ſeh<choice><am>ē</am><ex>en</ex></choice></w>
```

d) Tokenisierung und Abkürzung

In besonderen Fällen enthalten Abkürzungen klitisch angehängte Pronomina oder Negationspartikeln, sodass eine korrekte Tokenisierung erst nach der Auflösung der Abkürzung erfolgen kann. Solche Fälle

lösen wir, indem wir die <w>-Elemente in die Auflösung der Abkürzung setzen. Dazu allerdings müssen wir den gesamten Buchstabenkomplex, den die Abkürzung enthält, in der Orginalfassung als <abbr> erfassen, die Auflösung als <expan>; die Tokens stehen dann im Element <expan>.

Beispiel:

```
<choice>
    <abbr>moh<am>t̓</am></abbr>
    <expan> <w>moh<ex>t</ex></w> <w><ex>er</ex></w> </expan>
</choice>
```

e) Besondere Fälle: Hs. A (Heidelberg) verwendet als Abkürzung für *und* oft die Form u^{n}. Wir kodieren das, indem wir anstatt eines Abkürzungszeichens einfach mit <hi> ein hochgestellts *n* einsetzten:
<w><choice><am><hi rendition="hc:Superscript">n</hi></am><ex>nd</ex></choice></w>

23. Normalisierungen

Wir führen bei der Transkription bereits standardmäßige sprachliche Normalisierungen durch. Wir normalisieren konsonantisches *u/i* zu *v/j*, vokalisches *v/j* zu *u/i*, sowie konsonantisch-vokalisches *w* zu *wu*.

a) Ähnlich wie bei Abkürzungen werden Normalisierungen mit einem <choice>-Konstrukt erfasst.

b) Die Schreibung der Handschrift wird wiedergegeben im Element <orig>.

c) Die Normalisierung wird erfasst im Element <reg>.

Beispiele

vnd → und <w><choice><orig>v</orig><reg>u</reg></choice>nd</w>

vn̄ → und

<w><choice><orig>v</orig><reg>u</reg></choice><choice><am>n̄</am><ex>nd</ex></choice></w>

ivncfrowe → juncfrowe

<w><choice><orig>i</orig><reg>j</reg></choice><choice><orig>v</orig><reg>u</reg></choice>ncfrow
e</w>

wrden → wurden <w><choice><orig>w</orig><reg>wu</reg></choice>rden</w>

Framework: Insert Tags → regularize; *Shortcut:* shift + ctrl/cmd + R
(Normalisierung von Buchstaben mit Diakritika muss meist im Text-Modus erfolgen)

d) Achtung: Das Element <hei:initial> erlaubt kein <choice>-Konstrukt in seinem Inneren. Falls also Initialen normalisiert werden müssen, z.B. ***U**il*, muss das Element <hei:initial> sowohl im Element <orig> wie im Element <reg> wiederholt werden:

<w><choice><orig><hei:initial hei:color="Blue" hei:heightLines="4">U</hei:initial></orig>
<reg><hei:initial hei:color="Blue" hei:heightLines="4">V</hei:initial></reg></choice>il</w>

e) Abkürzung mit Normalisierung:
In manchen Fällen muss der Buchstabe, der das Abkürzungszeichen führt, normalisiert werden. Das geschieht durch ein doppeltes <choice>-Konstrukt, das der Abkürzung steht innerhalb der Elements <orig> der Normalisierungs-Konstrukts. Die Spatien im folgenden Beispiel dienen lediglich dazu, den Aufbau zu verdeutlichen:
v̄ → und:
<choice><orig> <choice><am>v̄</am><ex>vnd</ex></choice> </orig><reg>und</reg></choice>

f) Abkürzung in Versalien oder Rubrizierungen:
Versalien oder mit rotem Strich markiert Versanfangsbuchstaben, die mit dem Element <hi> getaggt werden, enthalten manchmal Abkürzungen, z.B. H̔re. Da bei Auflösung der Abkürzung nicht alle Buchstaben als hervorgehoben zu denken sind, sondern lediglich der Anfangsbuchstabe (Herre), muss das Element <hi> (ähnlich wie oben unter d bei der Normalisierung von Initialen) sowohl im Element <am> wie auch im Element <ex> wiederholt werden:
<choice><am><hi rendition="hc:Versal">H̔</hi></am><ex><hi rendition="hc:Versal">H</hi>er</ex>
</choice>

24. Abschnittszusammenfassungen

Der Iwein enthält keine Abschnitte, doch zumindest eine Handschrift (p = Paris) enthält Abschnittszusammenfassungen der Art "Hier kommt Herr Iwein an den Brunnen". Diese Stücke taggen wir zeilenweise mit dem Element <label> und dem Attribut ana="hc:SummaryLabel". Sollten diese Zusammenfassungen sich über mehr als eine Zeile erstrecken, wird vor dem <lb/> das Element <label> geschlossen und danach ein neues eröffnet.

25. @xml:id und @target

Elemente, die markiert werden sollen, um an anderer Stelle auf sie zu verweisen, werden mit einer @xml:id versehen. Der Wert des Attributs muss eine in der Datei einzigartige Zeichenkette sein, xml:ids dürfen sich also nicht wiederholen. Zur Einheitlichkeit wählen wir folgenden Aufbau von @xml:ids: "Handschrift_Element_Versnummer_laufendeNummer". So wäre z.B. eine @xml:id="A_w_765_1" sofort erkennbar als Handschrift A, Element 'Wort', Vers 765, erstes Wort des Verses. IDs, die sich nicht auf Bereiche des Textes beziehen, haben allerdings eine andere Struktur, => 3.

Auf @xml:ids wird verwiesen mit dem Attribut @target, dessen Wert die xml:id des Elements ist, auf das das Target verweist, mit einer Raute davor, z.B.: @target="#A_w_765_1".

26. Umstellungen

Für die Kodierung der Umstellungen von Wörtern und Wortgruppen ist es notwendig, dass die betroffenen Wörter jeweils mit einer @xml:id versehen sind. Die Textpassage wird zunächst so transkri-

biert, wie sie vor der Umstellung aussah. Direkt nach dem umgestellten Textsegment oder am Ende der Transkription der betroffenen Zeile (jedenfalls *vor* dem nächsten <lb/>) wird anschließend mit <listTranspose> die Umstellung dokumentiert. Das Kindelement <transpose> enthält eine Sequenz von <ptr>-Elementen, welche die umgestellten Wörter in der richtigen Reihenfolge (d.h. in der Reihenfolge nach der durchgeführten Umstellung) als Verweise auflisten.

Umstellungszeichen können in der Transkription mit <metamark ana="hc:TranspositionMark"> dokumentiert werden. Ihr Attribut @target verweist jeweils auf das Wort oder die Wörter, auf das bzw. auf die sich das Umstellungszeichen als Anweisung bezieht.

Beispiel:
Hs.: Der "auch "ſey ſelb ſo bewart →

```
<w>Der</w>
<metamark ana="hc:TranspositionMark" hei:placeRef="hc:AboveLine" target="#M_w_13_2">"
</metamark>
<w xml:id="M_w_13_2">auch</w>
<metamark ana="hc:TranspositionMark" hei:placeRef="hc:AboveLine" target="#M_w_13_3">"
</metamark>
<w xml:id="M_w_13_3">seŷ</w>
<w>ſelb</w><w>ſo</w><w>bewart</w>
<listTranspose>
  <transpose corresp="#M_w_13_2 #M_w_13_3">
    <ptr target="#M_w_13_3"></ptr>
    <ptr target="#M_w_13_2"></ptr>
  </transpose>
</listTranspose>
```

Bei ganzen Versen, die in 'verkehrter' Reihenfolge abgeschrieben und vom Schreiber durch Zeichen als in anderer Reihenfolge zu lesen markiert sind, transkribieren wir die Verse in der 'richtigen' Reihenfolge, also in der vom Schreiber intendierten. Die Nummerierung am <lb>-Element allerdings gibt die physische Abfolge des Originals wieder. Ein <listTranspose>-Element informiert über den Schreibprozess.

Beispiel

```
<l n="3399">
  <lb n="17"/>
    <metamark ana="hc:TranspositionMark" hei:placeRef="hc:ColumnMarginLeft">"</metamark>
    <w>Daz</w><w>er</w><w>den</w><w>ſin</w><w>hat</w><w>verlorn</w>
</l>
<l n="3400">
  <lb n="16"/>
```

```
    <metamark ana="hc:TranspositionMark" hei:placeRef="hc:ColumnMarginLeft">b</metamark>
    <w>Von</w><w>bezzern</w><w>zvhten</w><w>wart</w><w>geborn</w>
</l>
<listTransose>
 <transpose>
  <ptr target="#xpath(//l[@n='3400'])"></ptr>
  <ptr target="#xpath(//l[@n='3399'])"></ptr>
 </transpose>
</listTranspose>
```

27. Lagensignaturen und Reklamanten

Lagensignaturen und Reklamanten (sowie evtl. andere Phänomene außerhalb des Textes) kodieren wir mit dem Element <fw>, das ein Attribut @ana führt, welches folgende Wert haben kann: "hc:Catchword" (Reklamante), "hc:QuireSignature" (Lagensignatur), "hc:PageHeader" (Seitentitel), "hc:ColumnHeader" (Spaltentitel), "hc:PageFooter" (Seitentitel im Seitenfuß), "hc:ColumnFooter" (Spaltentitel im Spaltenfuß), "hc:PageNumeral" (Seitennummer), "hc:ColumnNumeral" (Spaltennummer).

Wir stellen die Elemente <fw> an das Ende der Seite, zu der sie gehören, also vor dem <pb> der folgenden Seite.

Lagensignaturen und Reklamanten stehen in den Handschriften in eigenen Zeilen, sie müssen also jeweils mit einem <lb/> begonnen werden. Dieses enthält nicht nur das übliche @n-Attribut, sondern auch ein Attribut @facs. Der Wert dieses Attributs ist ein Pointer, also ein targetartiger Verweis auf die entsprechende <zone>, die im Element <facsimile> steht. Dort muss nämlich, da sich Signaturen und Reklamanten außerhalb regulärer Spalten befinden, für jedes Vorkommen ein Seitenbereich (<zone>) angelegt werden. Die Verknüpfung zwischen dem Element <fw> und der <zone> im Element <facsimile> erfolgt über diesen Pointer. Das folgende Beispiel ist angepasst an das Beispiel oben (=> 3).

Beispiel:

```
<fw ana="hc:QuireSignature"><lb facs="#b_12v-b" n="1"/>
   <num value="2"><w>
      <choice><abbr><am>ıı?</am></abbr><expan><ex>secundus</ex></expan></choice>
    </w></num>
</fw>
```

28. Anmerkungen

Anmerkungen werden mit dem Element <note> getaggt, sowohl Anmerkungen der Handschriften wie editorische Anmerkungen. Das Element enthält das Attribut @ana. Für Anmerkungen, die im Textzeuge stehen, sind je nach Fall folgende Werte vorgesehen: "hc:Gloss" (Glosse), "hc:NotaBene" (Notazeichen), "hc:Manicula" (Zeigehand), "hc:Footnote" (Fußnote), "hc:Endnote" (Endnote). Für editorische Anmerkungen sind je nach Fall folgende Werte vorgesehen: "hc:Comment" (Kommentar), "hc:TextCriticalNote"

(textkritische Anmerkung), "hc:TranscriptionNote" (Anmerkung zur Transkription), "hc:WitnessesNote" (Anmerkung zu herangezogenen Textzeugen), "hc:LociNote" (Anmerkung zu Quellen, Parallelen und Nachwirkung), "hc:FontesNote" (Anmerkung über Quellen), "hc:SimiliaNote" (Anmerkung über Parallelen), "hc:TestimoniaNote" (Anmerkung über Nachwirkung), "hc:BiblicalNote" (Anmerkung über biblische Quellen). Für die Transkription kommt in der Regel nur "hc:TranscriptionNote" in Frage; typischerweise werden hier Phänomene der Handschrift notiert, die für den Textbestand irrelevant sind, z.B. zeitlich nicht bestimmbare Unterstreichungen oder Markierungen, Wiederholungen besonderer Eigenheiten, Verzierungen außerhalb des Schriftspiegels, Federproben.

Bei editorischen Anmerkungen sollte das Element <note> zusätzlich ein Attribut @target führen, mit dem auf das Wort oder auf den Vers verwiesen wird, auf die sich die Anmerkung bezieht. Bei Verweis auf ein Wort sollte dieses mit einer @xml:id versehen werden und das @target auf diese verweisen; bei Verweise auf einen Vers kannt auch ein xpath-pointer zum Einzatz kommen (=> 19).

29. Kommentar

In der xml-Datei können private Kommentare hinterlassen werden, die keinen Effekt auf die Formatierung haben. In der Regel dient dies zu Notizen für spätere Überarbeitung, z.B. Anmerkungen der Transkribenden für die Editoren. Solchen Kommentare werden Geschrieben innerhalb der Zeichen <!-- -->. Kommentare sollten hinter dem Abschluss des </l>-Elements stehen.

Framework: Insert Tags → comment