

Das können Sie auch!

capella-Dateien automatisch bearbeiten

Seit das binäre capella-Format (*.cap) durch das offene CapXML-Format ergänzt wurde, haben sich bekanntlich ganz neue Perspektiven für Tüftler und Programmierer eröffnet. Wir wollen hier einmal eine sehr einfache, aber dennoch nützliche Aufgabe erledigen, die direkt aus unserer Support-Praxis stammt. Als Werkzeug brauchen wir nichts als unseren Texteditor. Wir benötigen keinerlei Programmierkenntnisse – alles, was wir unternehmen, ist das Abändern von sich selbst erklärendem Text.

Die Aufgabenstellung

Ein capella-Anwender schreibt Noten für einen Verlag. Dabei hat er es häufig mit Generalbassbezeichnungen zu tun:

The image shows a musical score for three parts: Dessus 1, Dessus 2, and Basse Continue. The lyrics are "Lau - da, lau - da, lau - da Si - on sal - - va - -". The Basse Continue part has a red oval around the first four notes, with figured bass notation below them: 6, #, 6, 7. The figured bass notation for the first four notes is: 6, #, 6, 7. The figured bass notation for the last two notes is: 6 4 # 7, 5 2.

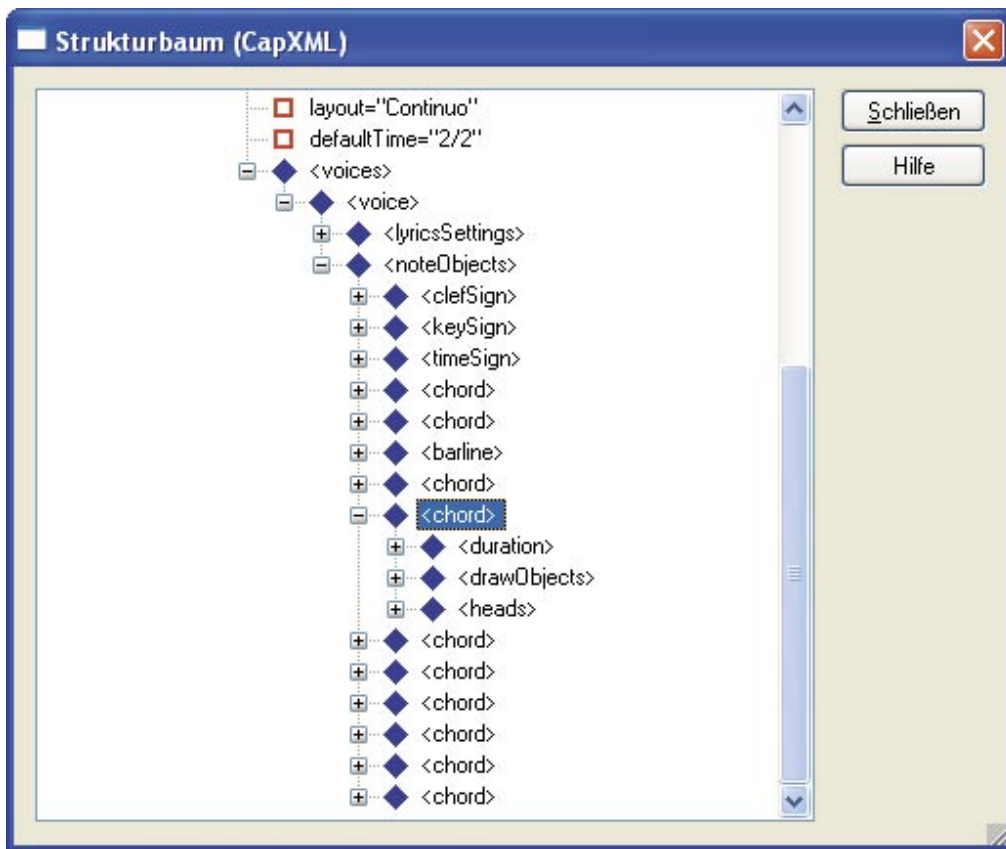
Die Generalbassbezeichnungen dieser Partitur finden wir in der Galerie **Generalbass**, so dass wir uns nicht die Mühe machen müssen, alle Zahlenkombinationen neu zu konstruieren. Wir verbinden die Galerie, die zum Lieferumfang von capella gehört, mit dem Menübefehl **Galerie - anfügen - Generalbass** mit unserer geöffneten Partitur. Jetzt können wir die Generalbasszeichen aus der Galerie nach Bedarf herauspicken.

Die Generalbassbezeichnung erscheint in der Schriftart **Arial** – das entspricht allgemeiner Praxis. Nun hat der Verlag, für den unser Anwender arbeitet, die Richtlinien geändert und verlangt für alle Publikationen eine **Times**-Schrift. Leider hat unser Anwender bereits 20 Partituren fertig gestellt. Mit konventionellen Mitteln müsste er zunächst eine neue Galerie anlegen, jedes Element einmal in der Partitur erzeugen und in dieser Galerie ablegen; danach müsste er in allen fertigen Partituren alle Generalbassbezeichnungen nacheinander löschen und durch die neuen aus seiner eigenen Times-General-bass-Galerie ersetzen.

Die Lösung: CapXML-Datei bearbeiten

Wir öffnen die Datei (Downloadquelle rechts im Kasten), stellen den Cursor vor die vierte Note in der Basstimme (Note b, Bezeichnung „6“) und wählen in capella den Befehl **Ansicht - Strukturbaum (CapXML)**.

capella zeigt uns die CapXML-Ansicht der Partitur und springt dabei zur aktuellen Note.



Auch ohne Programmierkenntnisse erkennen wir, dass unsere Note hier als `<chord>` bezeichnet wird und offensichtlich die Elemente `<duration>`, `<drawObjects>` und `<heads>` enthält. Es ist sicher auch kein Geheimnis, dass das Element `<drawObjects>` wohl Zeichenobjekte, also unsere Generalbassbezeichnung, darstellt. Durch Klick auf den Knoten davor und anschließend auf alle Knoten, die jeweils neu erscheinen, gelangen wir in die tiefsten Eingeweide dieses Textobjekts:

Werkzeuge für diesen Workshop

Generalbass

Die capella-Beispieldatei



http://www.capella.de/download/Workshop/Generalbass_01.cap

SciTE Texteditor

Komfortabler Editor mit Syntax-Hervorhebung bei CapXML (und weiteren Programmiersprachen)



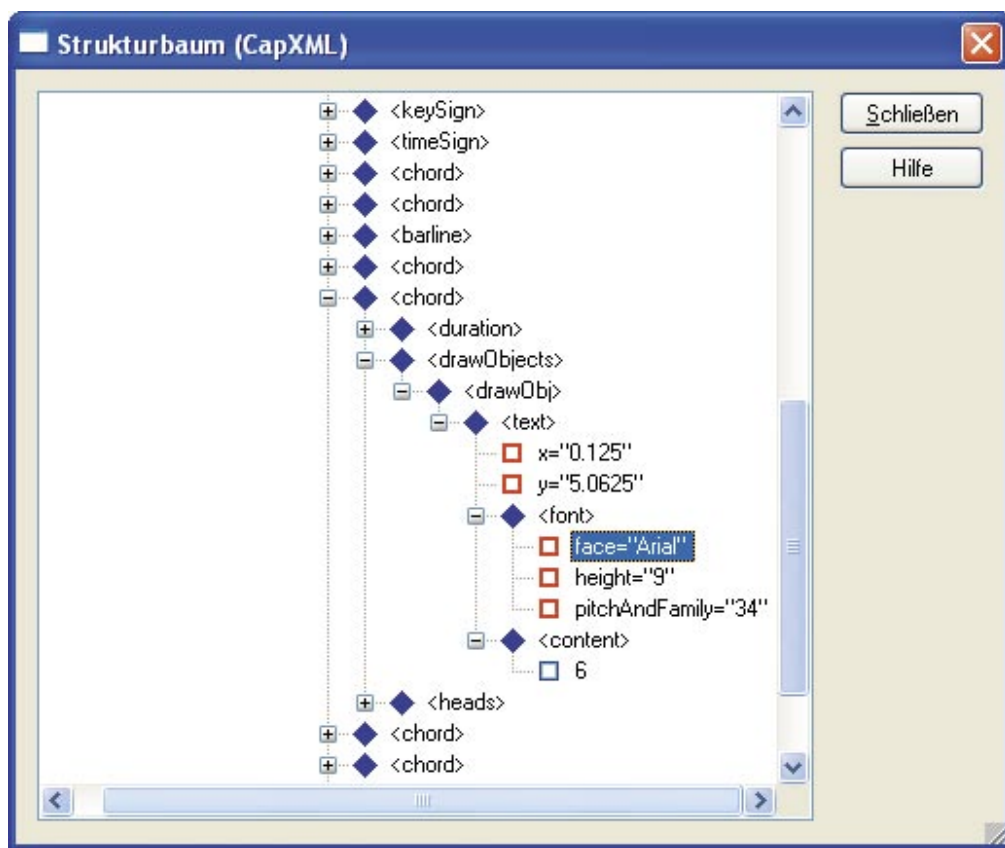
Kostenlos (Open Source)
<http://scintilla.sourceforge.net/SciTEDownload.html>

Reguläre Ausdrücke

Übersicht über die allgemein verwendeten Konventionen (siehe Seite 6)



http://de.wikipedia.org/wiki/Regul%C3%A4rer_Ausdruck



Mit ein wenig Fantasie können wir die Bedeutung aller Elemente ergründen. Auf jeden Fall erkennen wir, dass der Zeichensatz `` der Generalbassbezeichnung als Klartext `Arial` erscheint. Diese Bezeichnung wollen wir nun ändern, aber das tun wir in einem Editor.

CapXML-Datei abändern

- 🔧 Wir exportieren die Partitur als CapXML-Datei:
Datei - Exportieren - CapXML...
- 🔧 Wir schließen capella.
- 🔧 Wir gehen mit dem Windows-Explorer oder unserem Dateimanager in das Verzeichnis, in dem die Partitur liegt, und finden die Datei **Generalbass_01.capx**.
- 🔧 Wir ändern die Dateierweiterung von **capx** auf **zip**, so dass die Datei nun **Generalbass_01.zip** heißt.
- 🔧 Wir öffnen das Zip-Archiv mit unserem Entpacker (s. Kasten rechts). Das Zip-Archiv enthält die Datei **score.xml** – das ist die Partitur im CapXML-Klartext.
- 🔧 Mit komfortablen Werkzeugen wie dem Total Commander (s. rechts) können wir die Datei jetzt direkt mit dem Befehl **F4** bearbeiten. Anderenfalls müssen wir die Datei **score.xml** zunächst aus dem Archiv herausholen (z.B. mit Winzip) und irgend wohin kopieren.
- 🔧 Wir öffnen die Datei, falls noch nicht im vorigen Schritt geschehen, mit unserem Editor. Die folgenden Abbildungen haben wir mit **SciTe** erstellt, weil dieser Editor die Struktur der CapXML-Datei anschaulich in Farbe darstellt (s. nächste Seite).

Total Commander

Komfortabler Dateimanager – dem Windows-Explorer weit überlegen; mit einer Fülle von nützlichen Funktionen



Kostenlose Sharewareversion mit Möglichkeit des Erwerbs der Vollversion.

<http://www.ghisler.com/deutsch.htm>

AllaireHomeSite

HTML-Editor mit vielfältigen Suchfunktionen



Es gibt viele Anbieter. Die ältere Version 4.01 reicht für unsere Zwecke aus und ist u.a. bei Pearl erhältlich

<http://www.pearl.de>

Da die Angebote dort häufig wechseln, ist hier nur die Hauptseite verlinkt. Suchen Sie auf dem Portal von Pearl nach „Homepage“.

Impressum

capella-software GmbH
An der Söhrebahn 4
D-34320 Söhrewald
www.capella.de - info@capella.de

Tel. 05608-3923
Geschäftsführung: Edita Werner, M.A.
Handelsregister Kassel B 5589
StNr: 02623001254

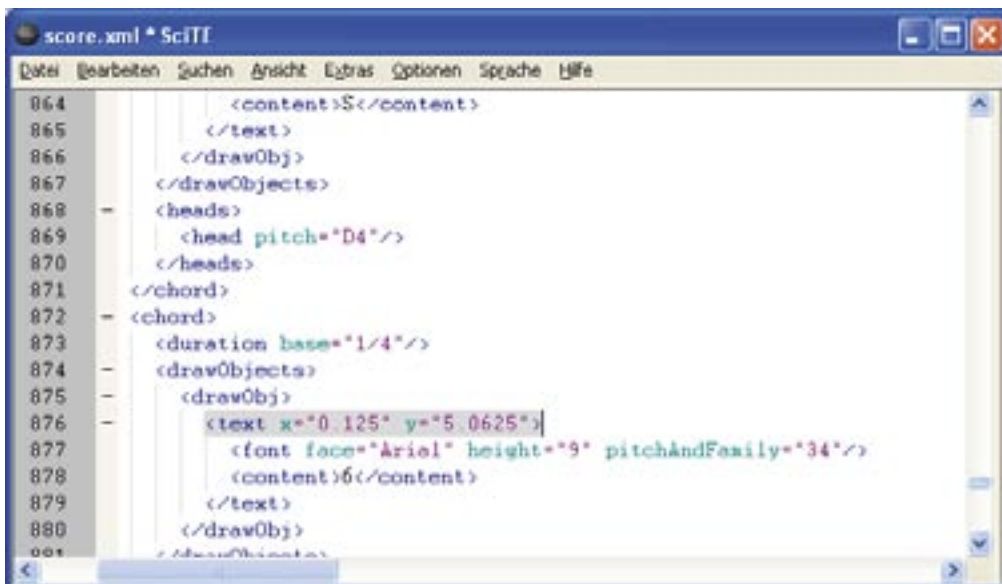
Alle Angebote in den **kontrapunkten** sind freibleibend. Es gelten unsere Geschäftsbedingungen: www.capella.de/agb.htm

Irrtum und Produktverbesserungen vorbehalten.



```
1 <?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
2 - <score xmlns="http://www.copella.de/CapXML/1.0">
3 - <|layout>
4 <pages left="10" top="10" right="10" bottom="15"/>
5 <distances>
6 <stafflines small="1.10" normal="1.52" pageObj="1.0"/>
7 <systems top="2" between="5"/>
8 </distances>
9 <instrumentNames align="right">
10 <font face="Times New Roman" height="8"/>
11 </instrumentNames>
12 <staves>
13 <staffLayout description="Sopran">
14 <notation defaultClef="treble">
15 <barlines mode="internal" from="3" to="7"/>
16 </notation>
17 <distances top="4" bottom="7"/>
18 <instrument name="Sopran 1" abbrev="" />
```

Im Editor sind alle Knoten geöffnet, auch die, die wir nicht sehen wollen; deshalb müssen wir die Stelle, die wir bearbeiten wollen, erst einmal suchen. Wollten wir den Begriff `Arial` suchen, würden wir erst einmal die Elemente in der Galerie selbst finden. Wir machen es uns einfacher und suchen exakt die Stelle, die wir oben in der Partitur betrachtet haben. Der Suchtext `<text x="0.125" y="5.0625">` bezeichnet die Koordinaten (den Platz) der Generalbassbezeichnung relativ zur Note und ist einmalig. Er führt uns zum ersten Generalbass-Element:



```
864 <content>S</content>
865 </text>
866 </drawObj>
867 </drawObjects>
868 <heads>
869 <head pitch="D4"/>
870 </heads>
871 </chord>
872 <chord>
873 <duration base="1/4"/>
874 <drawObjects>
875 <drawObj>
876 <text x="0.125" y="5.0625">
877 <font face="Arial" height="9" pitchAndFamily="34"/>
878 <content>6</content>
879 </text>
880 </drawObj>
881 </drawObjects>
```

Jetzt können wir den Text `Arial` eine Zeile tiefer durch `Times New Roman` ersetzen. Wir achten darauf, dass die Anführungszeichen erhalten bleiben. Die Zeile sieht jetzt in der farblichen Hervorhebung durch SciTE so aus:

```
<font face="Times New Roman" height="9" pitchAndFamily="34"/>
```

Natürlich wollen wir nicht nur diese eine, sondern alle Generalbassbezeichnungen abändern. Dazu markieren wir die folgenden Zeilen der CapXML-Datei bis zum Schluss und nehmen die Änderungen mit **Suchen - Ersetzen - In Markierung ersetzen** (oder sinngemäßer Befehl in Ihrem Texteditor) global vor.

Der Rest ist blitzschnell geschehen:

- Wir speichern die Datei unter dem gleichen Namen **score.xml**
- Wir packen die Datei ins ZIP-Archiv zurück und speichern das Archiv unter einem anderen Namen ab: **Generalbass_02.zip**
- Wir ändern die Dateierweiterung von zip nach capx: **Generalbass_02.capx**
- Wir öffnen die Datei von capella aus mit dem Befehl **Datei - Öffnen...**



```
score.xml * SciTE
Datei Bearbeiten Suchen Ansicht Extras Optionen Sprache Hilfe
912 - <text x="0.125" y="5.5625">
913 - <font face="Arial" height="9" pitchAndFamily="34"/>
914 - <content>6</content>
915 - </text>
916 - </drawObj>
917 - </drawObjects>
918 - <heads>
919 - <head pitch="B3">
920 - <alter step="-1"/>
921 - </head>
922 - </heads>
923 - </chord>
924 - <chord>
925 - <duration base="1/2"/>
926 - <drawObjects>
927 - <drawObj>
928 - <text x="0.28125" y="6.25">
929 - <font face="Arial" height="9" pitchAndFamily="34"/>
930 - <content>6
```

Jetzt erscheinen alle Ziffern in Times-Schrift (bitte benutzen Sie gegebenenfalls die Lupe Ihres Adobe-Readers).



Dessus 1
Lau - da, lau - da, lau - da Si - on sal - - va - -

Dessus 2
Lau - da Si - on, lau - da Si - on sal - - va - -

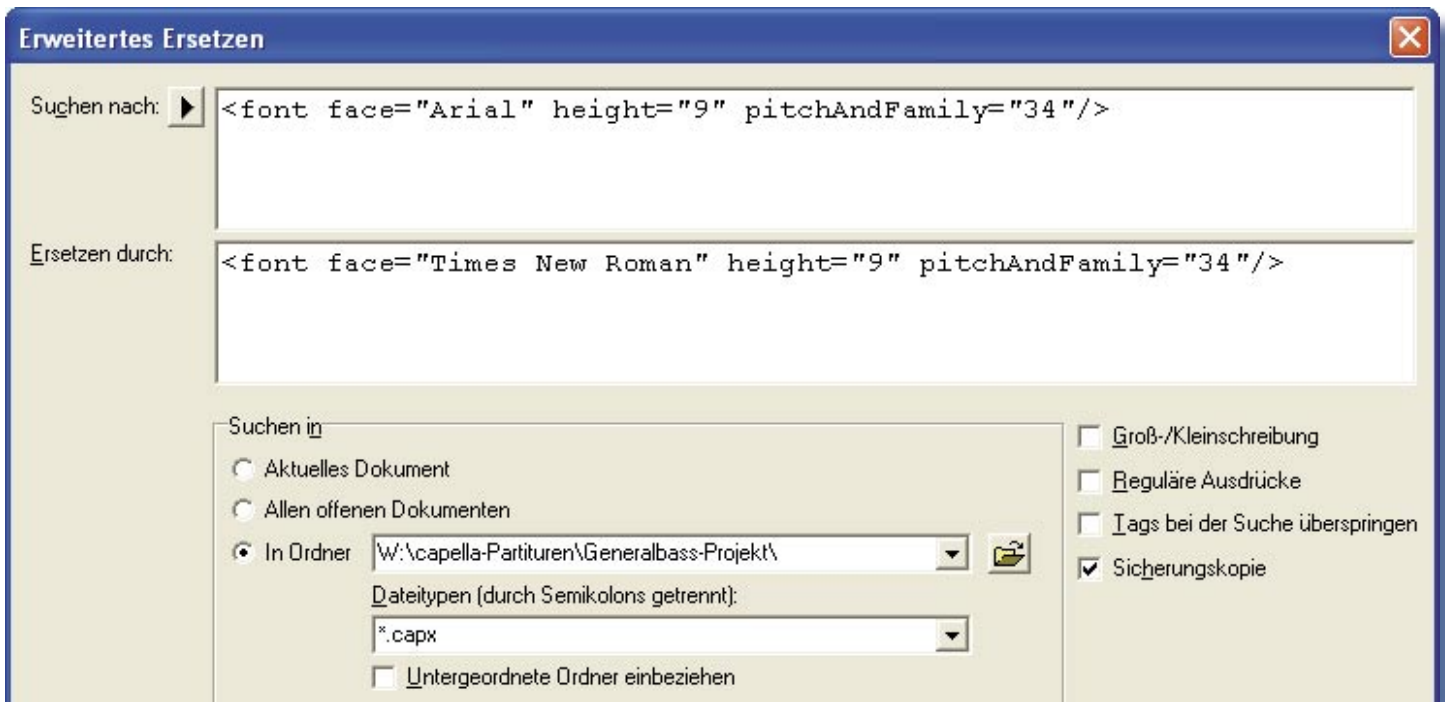
Basse Continue
6 # 6 6 4 # 7
3 2

Viele Dateien gleichzeitig bearbeiten

Manche Texteditoren können über Dateigrenzen hinweg suchen und ersetzen. Dazu können Sie z.B. den HTML-Editor Homesite verwenden. In der Version 4.0 bekommen Sie ihn praktisch geschenkt (s. rechts). Sie benötigen keine HTML-Kenntnisse, um damit CapXML-Dateien zu bearbeiten. Es folgt eine Kurzanleitung:

- Fassen Sie alle Dateien, die Sie bearbeiten wollen, in einem **Homesite-Projekt** zusammen.
- Erstellen Sie von allen Dateien Sicherheitskopien in einem anderen(!) Verzeichnis.
- Wählen Sie in Homesite den Befehl **Erweitertes Ersetzen** und geben Sie die Such- und Ersetzungsbegriffe ein.




Sie müssen bei der Auswahl der Begriffe sorgfältig sein, sonst kann es passieren, dass unerwünschte Fundstellen ebenfalls ersetzt werden.



Weiterungen

Vielleicht erscheint Ihnen das Ersetzen von Textattributen von Galerieinträgen sehr speziell. Es gibt aber viele weitere Fälle, in denen es sinnvoll sein kann, die Elemente Ihrer capella-Partitur nachträglich zu bearbeiten.

Hierzu schauen wir uns zum Schluss noch einmal den Beginn der CapXML-Datei an. Eingeschlossen durch die Wörter („Tags“) `<layout>` in Zeile 3 und `</layout>` in Zeile 40 steht eine Fülle von Begriffen, die uns, wenn wir sie erst einmal übersetzt haben, aus unserer Alltagspraxis mit capella sehr bekannt erscheinen. Nehmen wir an, Sie haben eine Partitur mit vielen Stimmenauszügen erstellt. Jetzt können Sie für alle Stimmenauszüge gleichzeitig

-  alle Seitenrandeinstellungen in einem Zug ändern (der Eintrag hierzu steht oben in der Zeile 4 der CapXML-Datei: `<pages left="10" top="10" right="10" bottom="15"/>`);
-  die Größe der Notenzeilen ändern (eine Zeile tiefer: `<staffLines small="1.18" normal="1.52" pageObj="1.8"/>`);
-  Lautstärke, Instrumentenbezeichnungen u.v.m. ändern.

Am besten nehmen Sie sich einmal eine Datei als Spielwiese vor – natürlich erstellen Sie vorher eine Sicherheitskopie - und probieren aus, was alles möglich ist. Fortsetzung nächste Seite.

Manche Editoren (wie SciTE und Homesite) erlauben das Ersetzen mit Hilfe **Regulärer Ausdrücke**.

In der Abbildung werden alle Schriftgrößen durch die einheitliche Größe 10 ersetzt. Eine Fundstelle zu den Konventionen der Regulären Ausdrücke finden Sie im Kasten auf S. 2.



Appetit auf mehr?

Das Ersetzen von Elementen mit dem Texteditor stellt die einfachste Stufe der Datei-manipulation dar. Am anderen Ende der Skala steht das Programmieren eines Skripts mit Python.

❶ Schreiben Sie in capella einmal in einer leeren Partitur ein paar Noten.

❷ Schauen Sie sich mit dem Menübefehl **Extras - Skript** einmal die Liste der installierten Skripte an.

❸ Wählen Sie das Skript **Bunte Noten**. - Die Noten werden abhängig von der Tonhöhe eingefärbt.

❹ Verlassen Sie capella und öffnen Sie in Ihrem Editor das Skript. Es liegt in Ihrem Installationsverzeichnis von capella im Unterverzeichnis `\scripts`

Der Quelltext ist keine CapXML-Datei, sondern Python Programmcode. Aber auch hier erkennen Sie schnell, was gemeint ist. Sie können die Farbwerte sehr einfach abändern, indem Sie die Zahlen in Klammern nach `Color.RGB(..., ..., ...)` ändern; Werte zwischen 0 und 255 sind erlaubt.

Die Zeilen darunter `for obj in activeScore()` usw. sind schwieriger zu lesen. Mit etwas Fantasie können Sie jedoch erahnen, dass das Skript alle Objekte der aktiven Partitur (`score`) durchgeht und, wenn die Objekte Noten (`chord`) und Köpfe (`head`) sind, die Köpfe bunt färbt. Die Anleitung, welche Farbe zu verwenden ist, holt der Programmcode aus der Liste oben mit Namen `C`.

Ein weiteres Beispiel, das etwas schwieriger zu lesen ist, dafür aber unsere Schriftgrößenersetzter auf sehr elegante Weise erledigt, ist das Skript rechts. Damit es seine Arbeit machen kann, erzeugt es aus der capella-Datei eine CapXML-Datei (`tempFile1`), wandelt diese nach `tempFile2` und konvertiert abschließend wieder nach `*.cap`.

Wenn Ihnen das Lesen der Quelltexte keine Mühe macht und Sie dabei Appetit auf mehr bekommen, dann können Sie sich an einem verregneten Herbstabend einmal mit den Grundlagen von Python befassen. Vielleicht wird so aus Ihnen gar ein Skripte-Verfasser?! Die Skript-Autoren haben übrigens eigene Homepages und helfen Einsteigern gern. Unter www.capella.de finden Sie den Weg zu ihnen.

```
# zwölf Farben für die chromatischen Töne der Oktave:
-C = (Color.RGB(255, 0, 0), # rot
      Color.RGB(255,128, 0), # orange
      Color.RGB(255,192, 0), # orangegelb
      Color.RGB(255,255, 0), # gelb
      Color.RGB(160,255, 0), # gelbgelbgrün
      Color.RGB(128,255, 0), # gelbgrün
      Color.RGB( 0,224, 0), # grün
      Color.RGB( 0,112,128), # blaugrün
      Color.RGB( 0, 64,192), # grünblau
      Color.RGB( 0, 0,255), # blau
      Color.RGB( 96, 0,160), # violett
      Color.RGB(224, 0, 80)) # rotviolett

- for obj in activeScore().noteObjs():
-     if obj.isChord() and obj.nHeads() == 1:
-         pitch = obj.head(0).chromaticPitch() % 12
-         obj.setColor(C[pitch])
```

```
# -*- coding: ISO-8859-1 -*-
from caplib.capDOM import ScoreChange
import tempfile

class TestChange (ScoreChange):
    def changeElement(self, el):
        # Bei allen "font"-Elementen, die Kinder von
        # "text"-Elementen sind,
        # die Schrift um 50% vergrößern:
        if el.tagName == "font" and el.parentNode.
            tagName == "text":
            alt = float(el.getAttribute("height"))
            neu = 1.5 * alt
            el.setAttribute("height", str(neu))

if activeScore():
    activeScore().registerUndo("Texte vergrößern")
    tempFile1 = tempfile.mktemp(".capx")
    tempFile2 = tempfile.mktemp(".capx")
    activeScore().write(tempFile1)
    TestChange(tempFile1, tempFile2)
    activeScore().read(tempFile2)
    os.remove(tempFile1)
    os.remove(tempFile2)
```