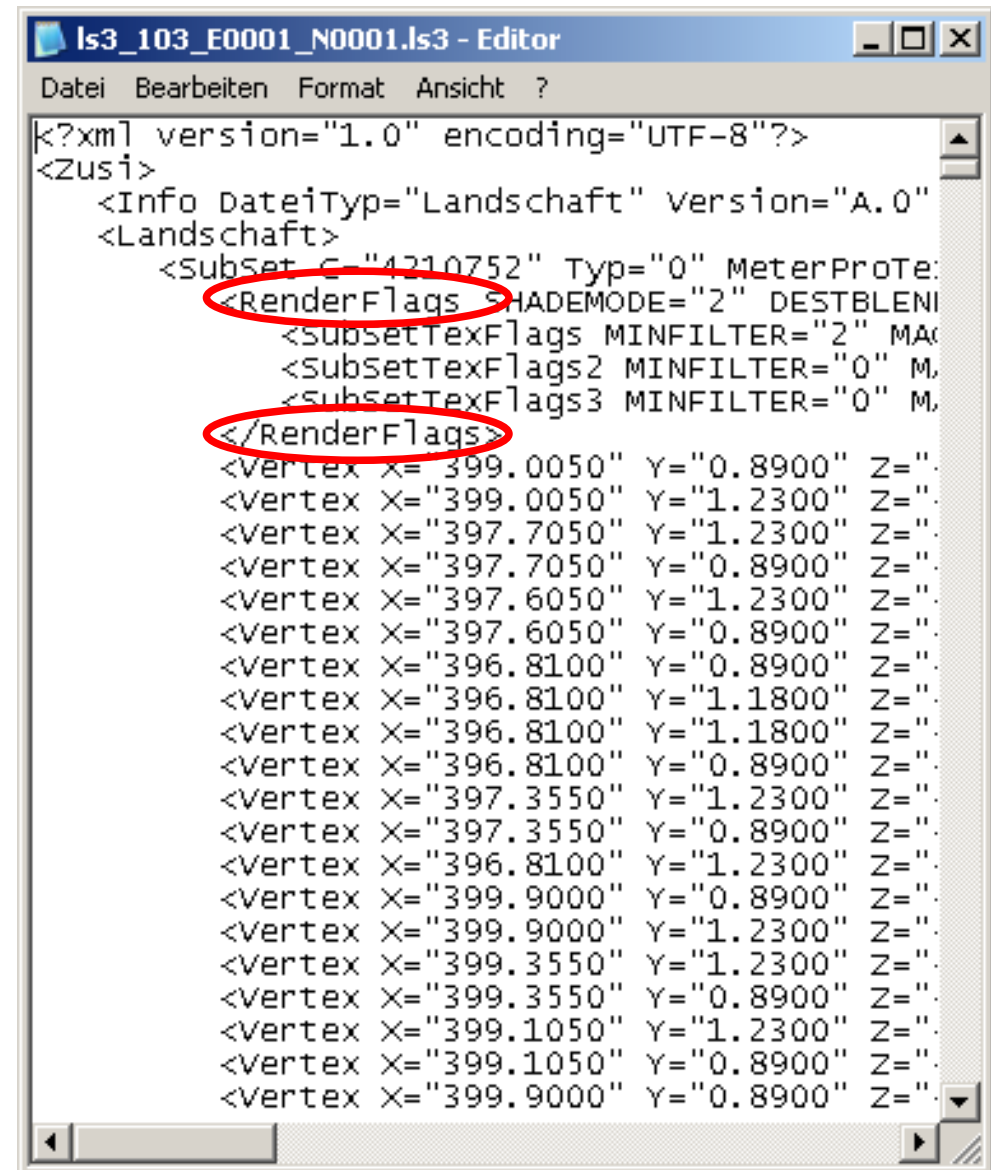


XML-Format

- „xml“: extensible markup language
- Datenformat aus Knoten und Attributen in einer Hierarchie (wie z.B. Dateien im Explorer)
- Format ist selbstbeschreibend
- Daten werden über einen Text-Bezeichner identifiziert (Name=“Wert“, z. B. XKoordinate = “36.34“)
- Dateiformat ist ASCII, also mit jedem Texteditor zu öffnen
- xml-Editoren erleichtern die Handhabung durch strukturierte Darstellung
- xml-Editoren lesen die Syntax, ohne den Inhalt „zu verstehen“
- Verschiedene Editorkonzepte werden jetzt kurz vorgestellt

xml-Datei: Quelltext

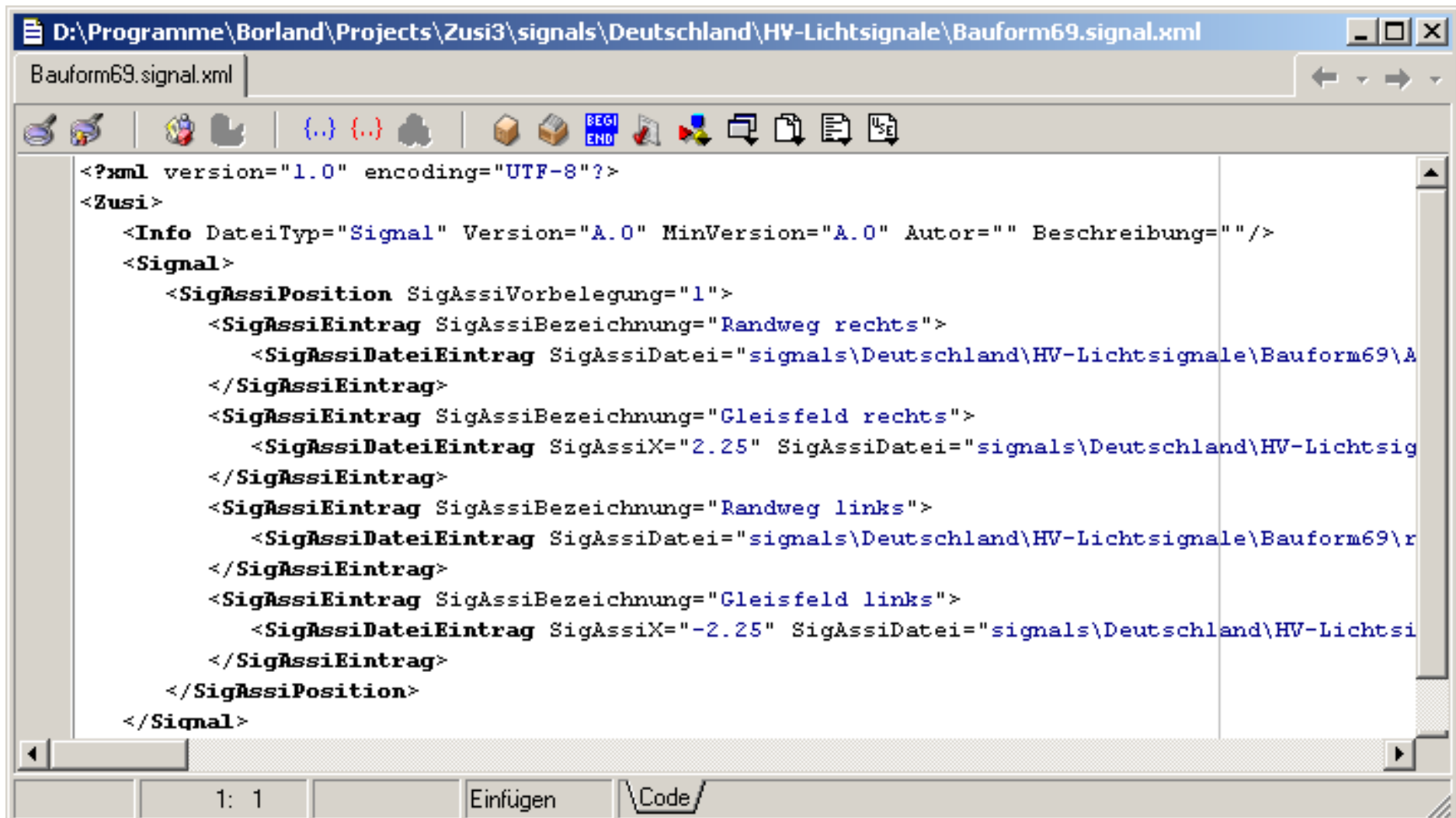
- Darstellung: MS-Notepad
- Knoten nur mit Attributen definiert durch:
`<name [Attribute...] />`
- Knoten mit Unterknoten:
`<name> [Attribute...]
[Unterknoten...]
</name>`
- Attribute definiert mit
`Name = "Wert"`, sind
einem Knoten zugeordnet
- Einrückung und
Zeilenumbrüche formal
nicht nötig, erhöhen aber
die Lesbarkeit im
Texteditor



```
ls3_103_E0001_N0001.ls3 - Editor
Datei Bearbeiten Format Ansicht ?
<?xml version="1.0" encoding="UTF-8"?>
<Zusi>
  <Info DateiTpe="Landschaft" Version="A.0">
    <Landschaft>
      <Subset C="4210752" Typ="0" MeterProTe:
        <RenderFlags SHADEMODE="2" DESTBLENI
          <SubsetTexFlags MINFILTER="2" MA
            <SubsetTexFlags2 MINFILTER="0" M
              <SubsetTexFlags3 MINFILTER="0" M
            </RenderFlags>
          <Vertex X="399.0050" Y="0.8900" Z="
            <Vertex X="399.0050" Y="1.2300" Z="
            <Vertex X="397.7050" Y="1.2300" Z="
            <Vertex X="397.7050" Y="0.8900" Z="
            <Vertex X="397.6050" Y="1.2300" Z="
            <Vertex X="397.6050" Y="0.8900" Z="
            <Vertex X="396.8100" Y="0.8900" Z="
            <Vertex X="396.8100" Y="1.1800" Z="
            <Vertex X="396.8100" Y="1.1800" Z="
            <Vertex X="396.8100" Y="0.8900" Z="
            <Vertex X="397.3550" Y="1.2300" Z="
            <Vertex X="397.3550" Y="0.8900" Z="
            <Vertex X="396.8100" Y="1.2300" Z="
            <Vertex X="399.9000" Y="0.8900" Z="
            <Vertex X="399.9000" Y="1.2300" Z="
            <Vertex X="399.3550" Y="1.2300" Z="
            <Vertex X="399.3550" Y="0.8900" Z="
            <Vertex X="399.1050" Y="1.2300" Z="
            <Vertex X="399.1050" Y="0.8900" Z="
            <Vertex X="399.9000" Y="0.8900" Z="
```

xml-Editor mit Syntaxhervorhebung

- Datei wird als Quelltext dargestellt, aber die Attribute, Werte und Knoten werden farblich hervorgehoben. Beispiel Delphi-IDE:



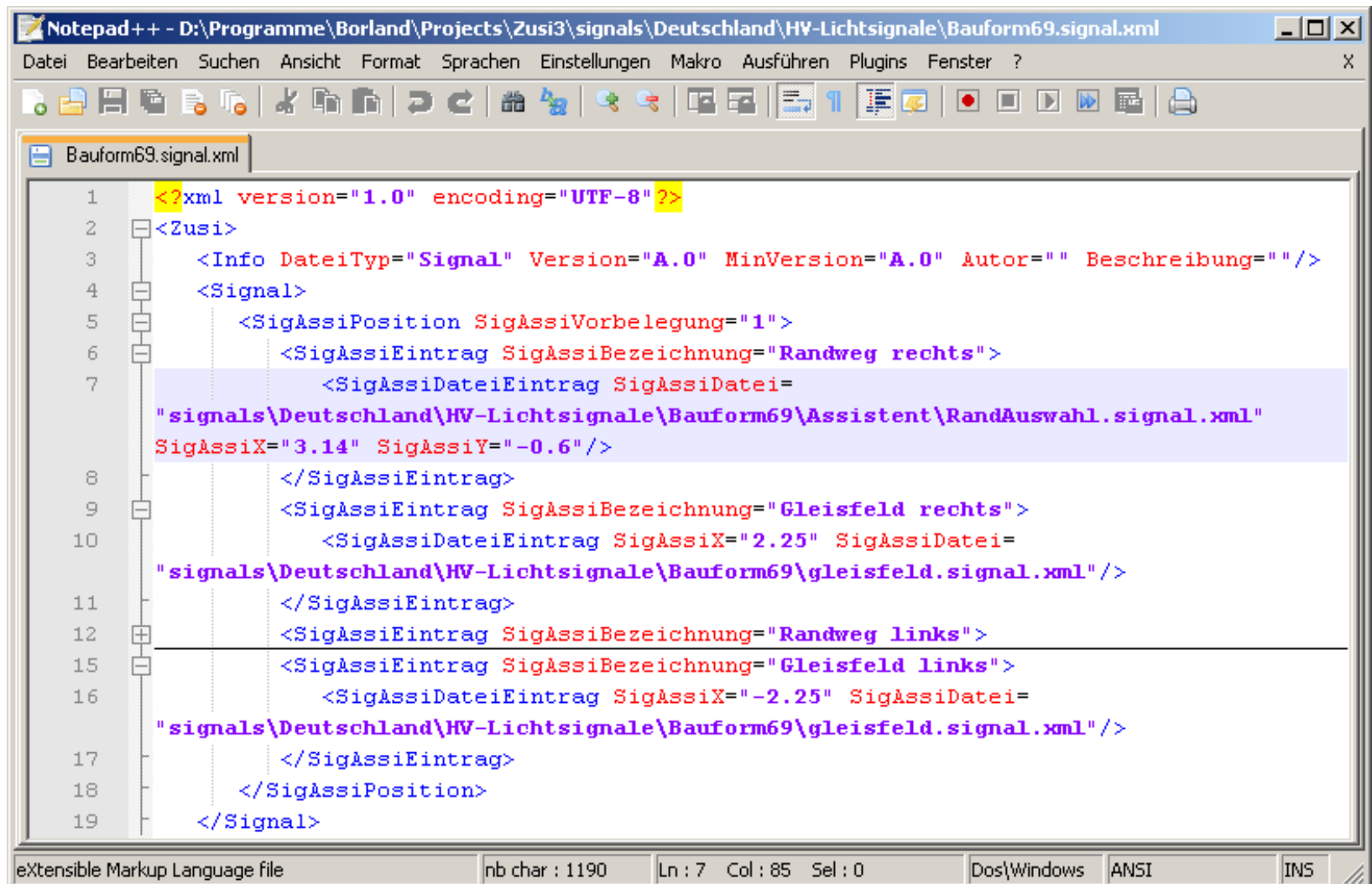
The screenshot shows the Delphi IDE with a file named 'Bauform69.signal.xml' open. The XML content is displayed with syntax highlighting: tags are in black, attributes and values are in blue, and element names are in black. The code is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<Zusi>
  <Info DateiTyp="Signal" Version="A.0" MinVersion="A.0" Autor="" Beschreibung="" />
  <Signal>
    <SigAssiPosition SigAssiVorbelegung="1">
      <SigAssiEintrag SigAssiBezeichnung="Randweg rechts">
        <SigAssiDateiEintrag SigAssiDatei="signals\Deutschland\HV-Lichtsignale\Bauform69\A
      </SigAssiEintrag>
      <SigAssiEintrag SigAssiBezeichnung="Gleisfeld rechts">
        <SigAssiDateiEintrag SigAssiX="2.25" SigAssiDatei="signals\Deutschland\HV-Lichtsig
      </SigAssiEintrag>
      <SigAssiEintrag SigAssiBezeichnung="Randweg links">
        <SigAssiDateiEintrag SigAssiDatei="signals\Deutschland\HV-Lichtsignale\Bauform69\r
      </SigAssiEintrag>
      <SigAssiEintrag SigAssiBezeichnung="Gleisfeld links">
        <SigAssiDateiEintrag SigAssiX="-2.25" SigAssiDatei="signals\Deutschland\HV-Lichtsi
      </SigAssiEintrag>
    </SigAssiPosition>
  </Signal>
</Zusi>
```

The IDE interface includes a menu bar, a toolbar with icons for search, save, and other functions, and a status bar at the bottom showing '1: 1' and 'Einfügen'.

xml-Quelltexteditor mit Hierarchie

- Daten werden als Quelltext mit farblicher Kennung angezeigt und können wie im Explorer reduziert werden. Beispiel Notepad++:



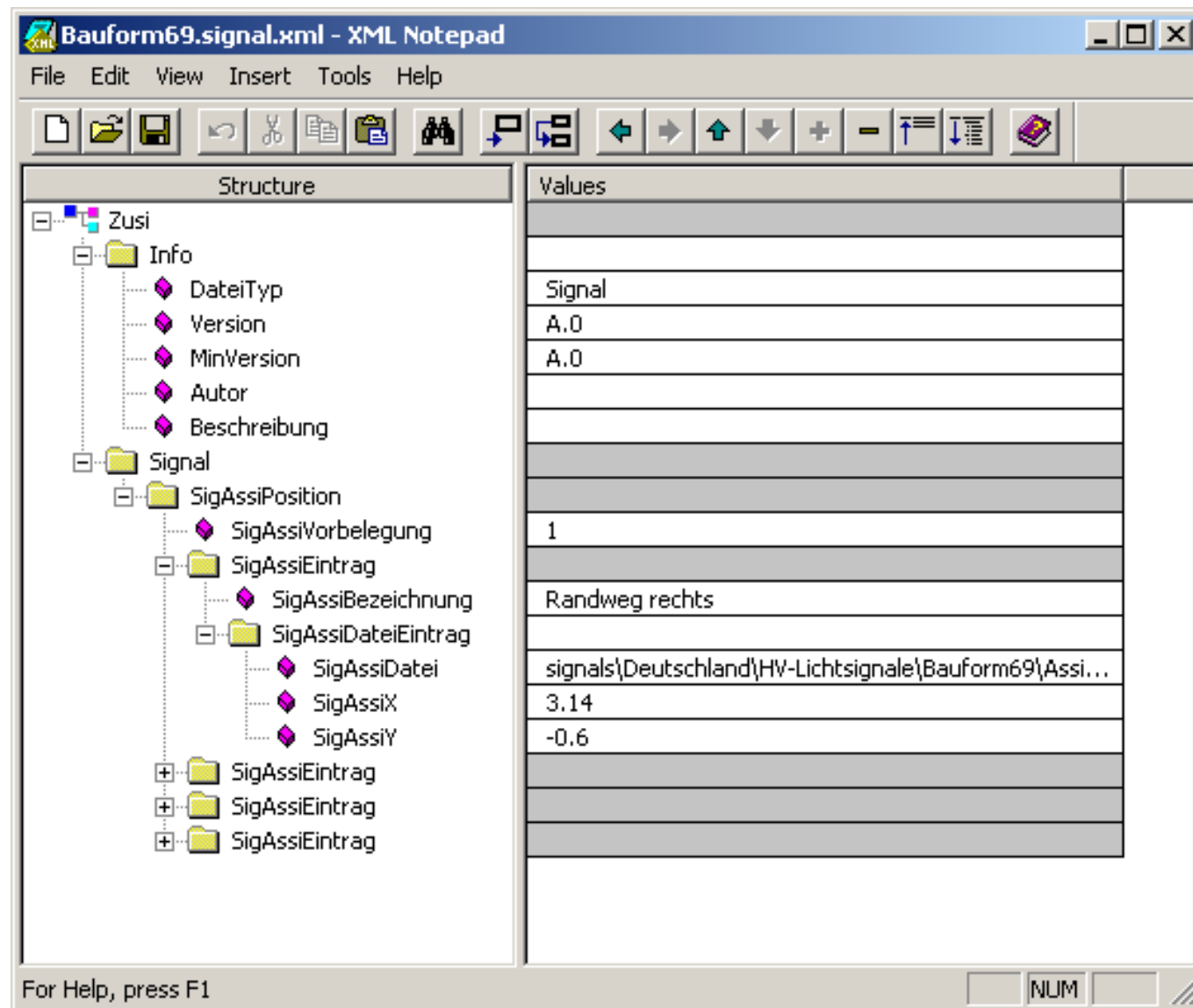
The screenshot shows the Notepad++ application window with the file `Bauform69.signal.xml` open. The interface includes a menu bar, a toolbar, and a tab bar. On the left, a hierarchical tree view displays the XML structure with expandable nodes. The main editing area shows the XML code with syntax highlighting: XML tags are in blue, attributes and values are in red, and text content is in black. The code is as follows:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Zusi>
3      <Info DateiTyp="Signal" Version="A.0" MinVersion="A.0" Autor="" Beschreibung="" />
4      <Signal>
5          <SigAssiPosition SigAssiVorbelegung="1">
6              <SigAssiEintrag SigAssiBezeichnung="Randweg rechts">
7                  <SigAssiDateiEintrag SigAssiDatei=
8                      "signals\Deutschland\HV-Lichtsignale\Bauform69\Assistent\RandAuswahl.signal.xml"
9                      SigAssiX="3.14" SigAssiY="-0.6"/>
10             </SigAssiEintrag>
11             <SigAssiEintrag SigAssiBezeichnung="Gleisfeld rechts">
12                 <SigAssiDateiEintrag SigAssiX="2.25" SigAssiDatei=
13                     "signals\Deutschland\HV-Lichtsignale\Bauform69\gleisfeld.signal.xml" />
14             </SigAssiEintrag>
15             <SigAssiEintrag SigAssiBezeichnung="Randweg links">
16                 <SigAssiDateiEintrag SigAssiX="-2.25" SigAssiDatei=
17                     "signals\Deutschland\HV-Lichtsignale\Bauform69\gleisfeld.signal.xml" />
18             </SigAssiEintrag>
19         </SigAssiPosition>
20     </Signal>
```

The status bar at the bottom indicates the file is an "eXtensible Markup Language file", contains 1190 characters, and the cursor is at line 7, column 85.

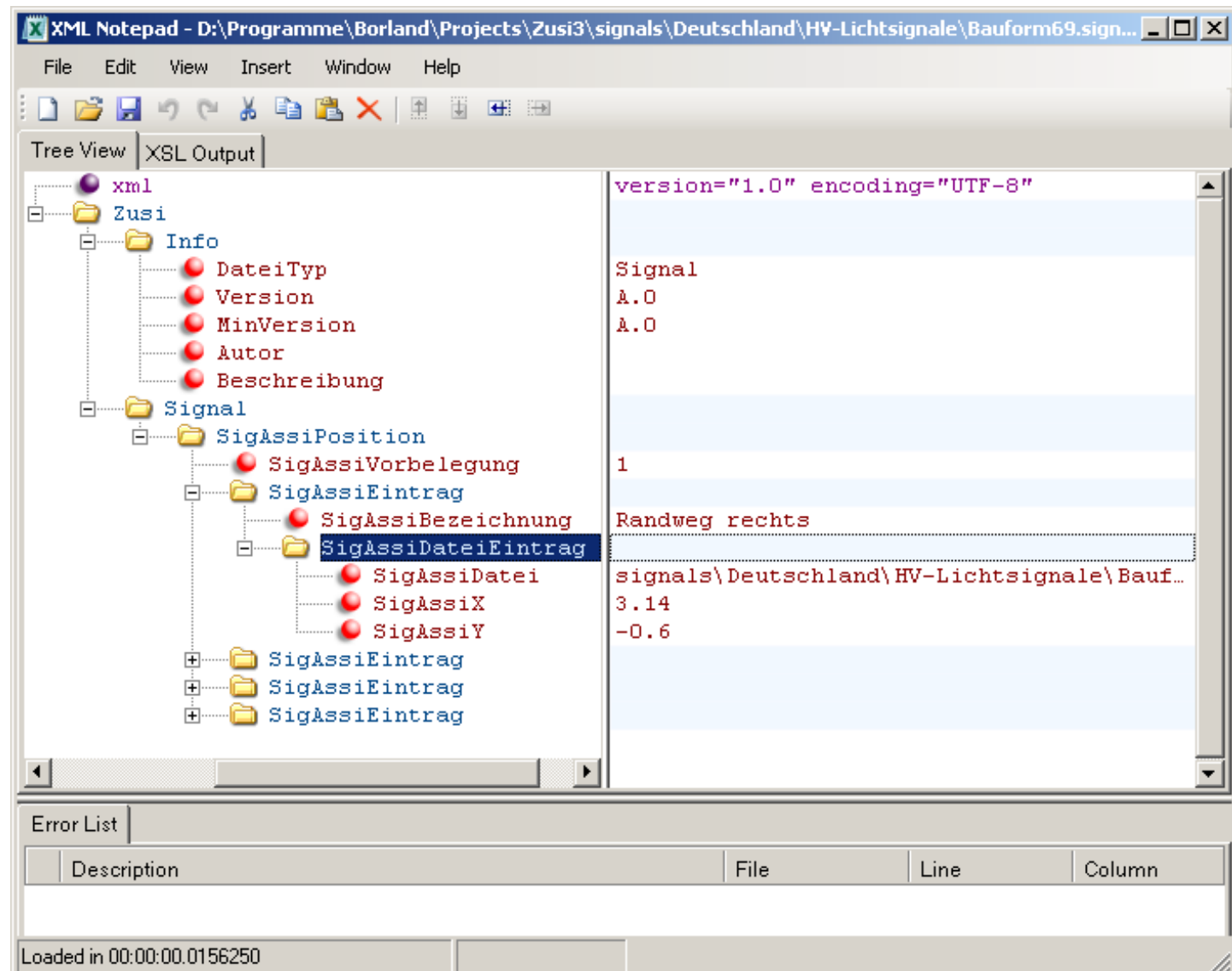
xml-Editor mit Eingabeoberfläche

- Im Quellcode muß nicht mehr editiert werden. Beispiel: MS-xml-Notepad 1.5 beta (1998):



Aktuelles MS-xml-Notepad

- 2007 erschien dann die endgültige MS-Notepad-Version als xml-Notepad 2007:

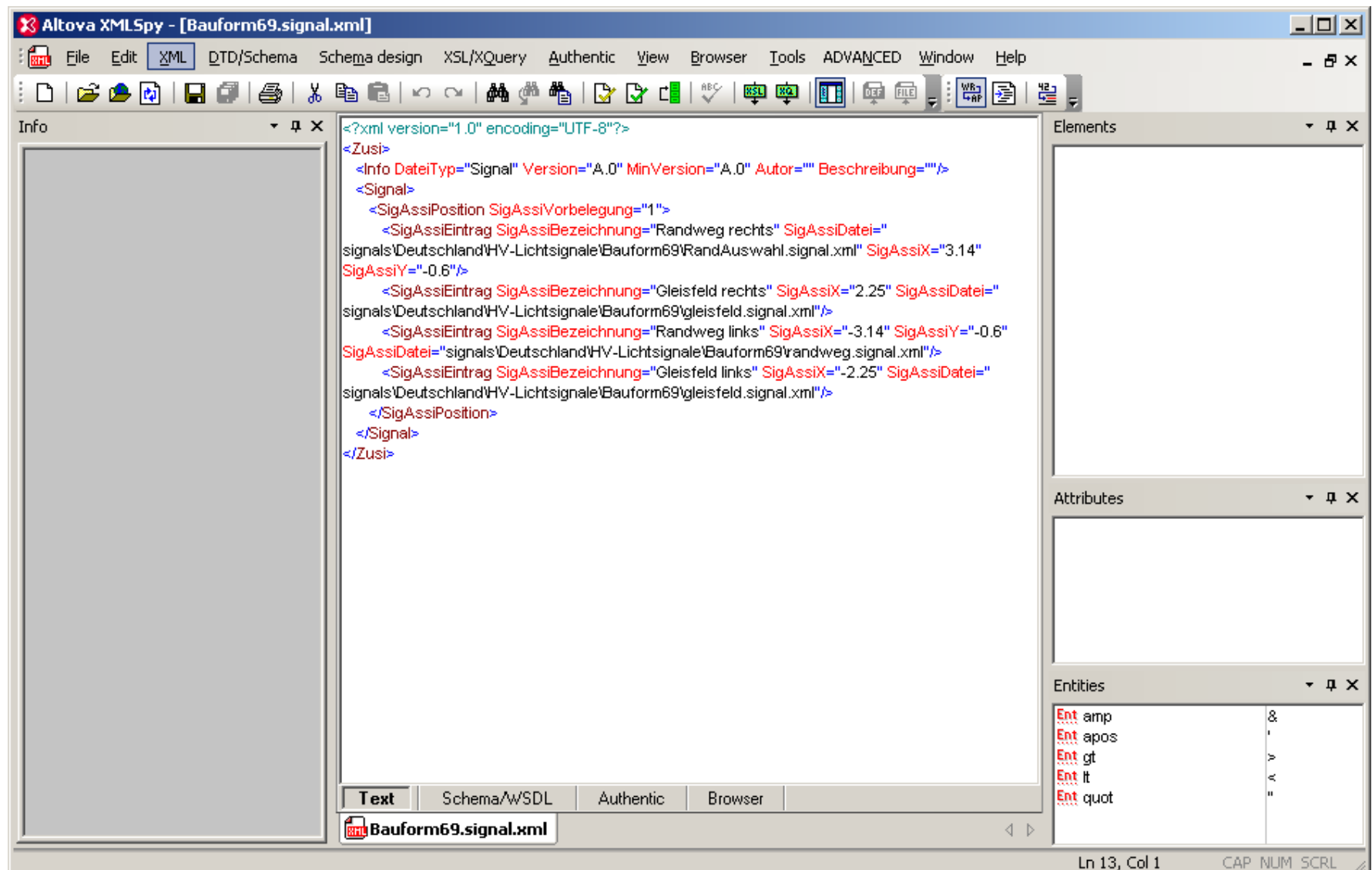


xml-Vorteile

- Dateiformat ist standardisiert: Für Einlesen und Speichern müssen keine eigenen Routinen programmiert werden (z.B. Zusi-Tools)
- Format ist ohne Konflikte erweiterbar, so daß sogar neuere (eigentlich unbekannte) Dateiformate gelesen werden können, wobei nur die neuen Attribute ignoriert werden
- Kaum Dokumentation nötig, da das Format selbstbeschreibend ist
- Es kann ein Schema des spezifischen (Zusi-)Formats erstellt werden, womit Editoren (wie Notepad 2007) sogar den Inhalt einer xml-Datei auf korrekte Werte überprüfen können. Es wird dabei in der Schema-Datei für jeden Knoten definiert, welche Unterknoten, Attribute und Werttypen zulässig sind

Profi-Editor

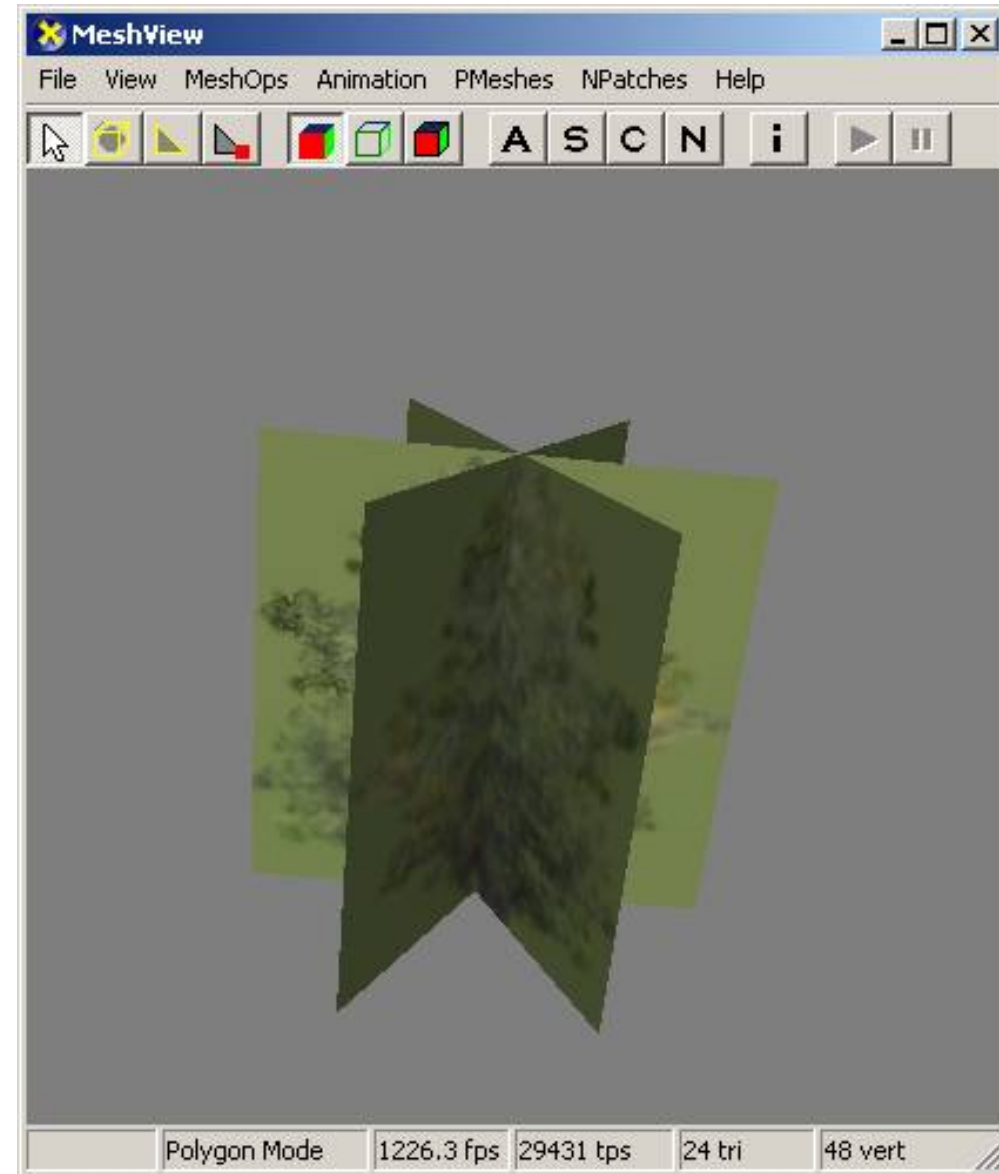
- Professioneller Editor mit Schema-Erstellung usw.: Altova XML-Spy (Basisversion kostenlos)



Mesh-Viewer aus DirectX-SDK

- Referenz-Darstellung für .x-Dateien
- Anwendung z.B. Test auf korrektes Dateiformat
- .x-Datei-Hierarchie
- Animation darstellbar
- Normalenvektoren
- Keine Transparenz
- Umwandlung ASCII/binär .x-Datei
- Erzeugung einfacher Körper

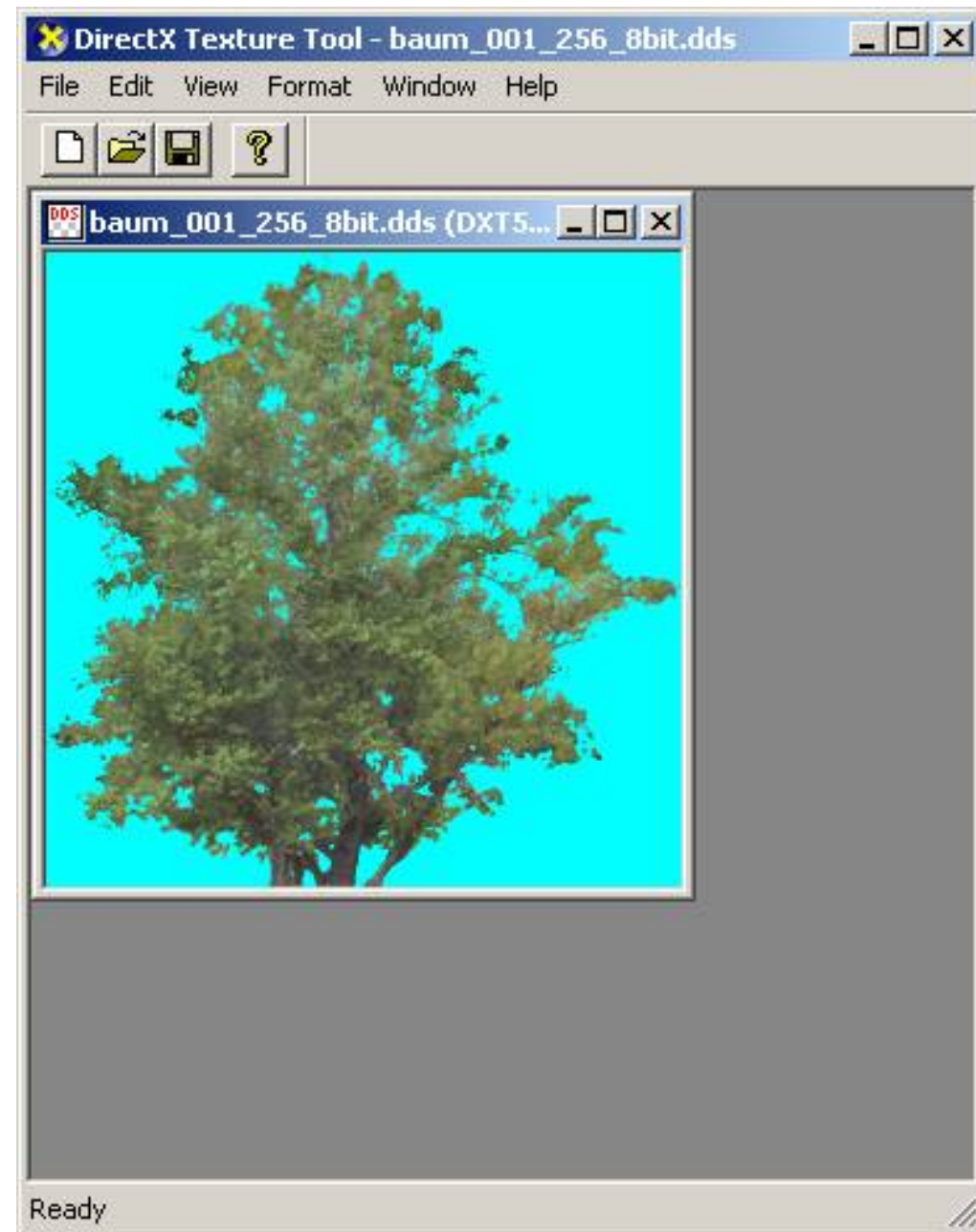
Dateiname: mview.exe



Texture-Tool aus DirectX-SDK

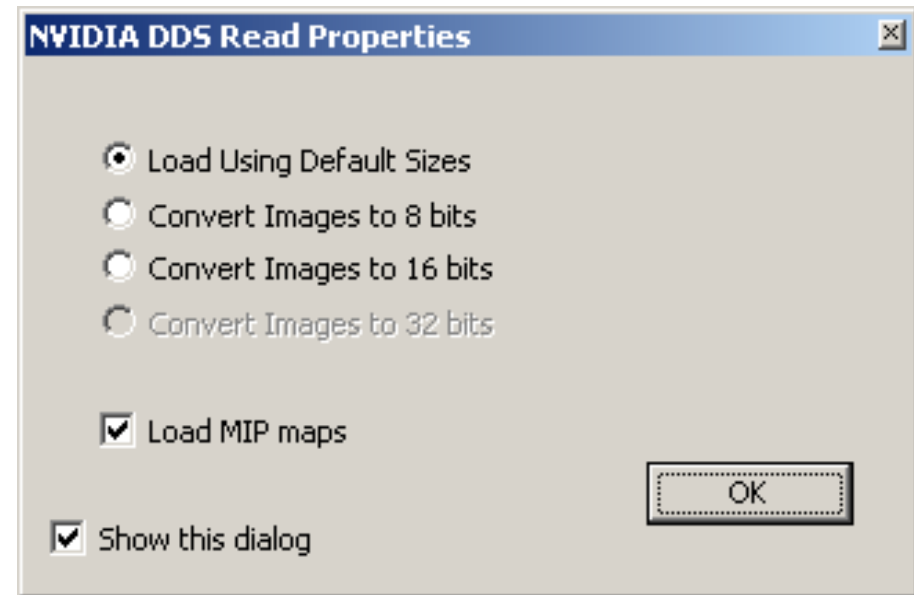
- Import gängiger Grafikformate
- Export als dds
- In dds-Unterformate umwandeln
- Alphakanal erzeugen bzw. dazuladen
- Mipmaps einzeln anzeigen
- Mipmaps erzeugen oder einzeln laden

Dateiname: DxTex.exe

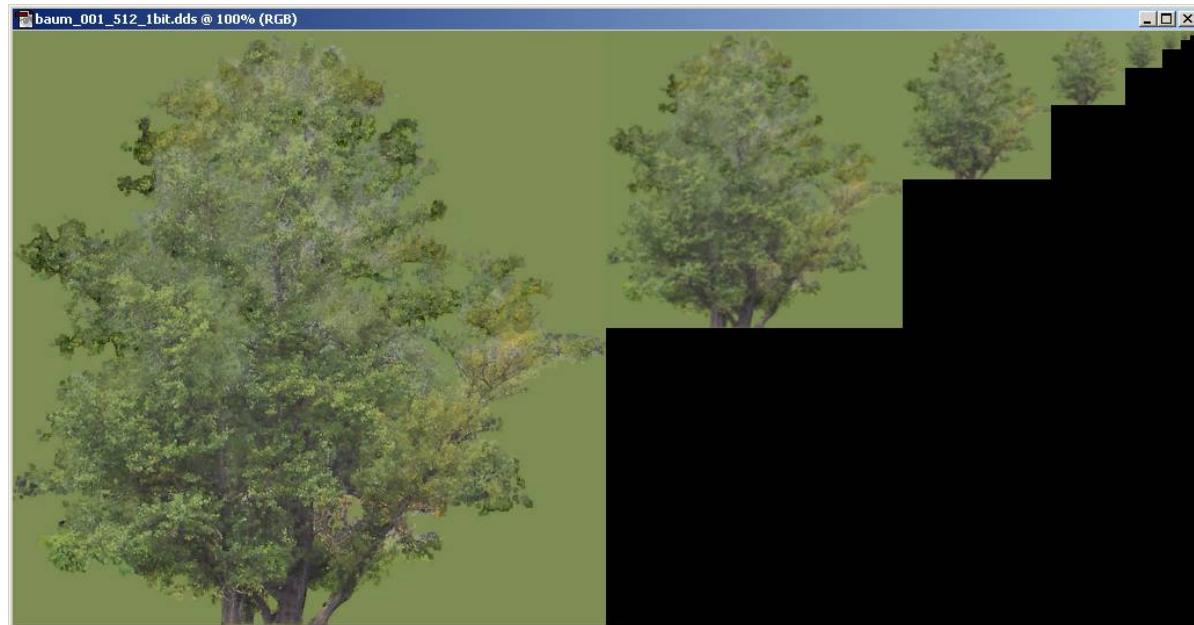


Adobe-Photoshop-dds-Plugin

- Beim Öffnen einer dds-Datei können die in der dds-Datei enthaltenen Mipmaps geladen werden



- Anzeige und Bearbeitung aller Mipmaps in einem Bild



Adobe-Photoshop dds-Export

- „Speichern unter“: dds
- DXT1- oder DXT3-Unterformat
- 2D Texture
- Use Existing Mipmaps speichert die Mipmaps wie vorhanden
- Generate Mipmaps erzeugt die Mipmaps aus dem Bildinhalt (in diesem Beispiel unsinnig, da Mipmaps schon vorhanden)
- Rest kann in Standardeinstellung bleiben

