Remarks on Visualizing BNF Grammars using Interactive SVG

August 27-30 2014

Winchester, England

Visual Storytelling

Jürgen Höhe
CreativeCologne UG

Köln Germany

Web Presentaion

Abstract

One and a half decades ago no dynamic and interactive graphics could be rendered in a browser. Dynamics had to simulated by generating pictures at the server side. This has been changed when SVG (Scalable Vector Graphics) came up.

This paper shows how EBNF (Extended Backus Naur Form) – a metalanguage to describe computer languages – can be turned into diagrams using SVG. The generated graphs can be modified interactively. Using this technique the main drawback of EBNF – readibility and thumbing through rules – can be compensated.

Some general aspects of diagrams are discussed.

Introduction

In Germany one needs a theory before any practical work can start. The search for „theory of diagrams“ brought some refrences to very abstract mathematical explanation and a link to a biannual, international and interdisciplinary conference. The papers found did not answer the simple and basic question why diagrams can show interrelationships better than written text.

Wikipedia lists more lists more than hundred different diagram types. There are strong doubts whether all of them can be covered by a common theory. The areas and purposes of the diagrams seem to be too different.

Here a user-friendly approach is preferred by raising some critical questions concerning specific diagrams. Here are some of them related to interactive syntax diagrams:

- is visualization more comprehensive than written text?
- is it precise?
- is it intuitive?
- does its interaction provide a better understanding?
- is it worthwhile (cost effectiveness)?
- is it more than a toy?
- what are the limits?
- is the technical base – Javascript, HTML plus embedded SVG - sufficient?

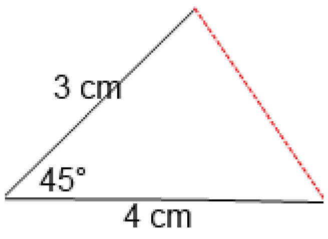
During the presentation of interactive BNF diagrams some of these questions will be answered.

Examples of Useful Diagrams

The world of today is full of diagrams. Traffic signs rule our mobile world.

A traffic sign represents in the format of a diagram a rule of traffic. It would not be practical to replace it by the corresponding paragraph.

A triangle could be defined by three figures: length of the left side of an angle, size of an angle in degrees and the length of the right side: 3, 45, 4



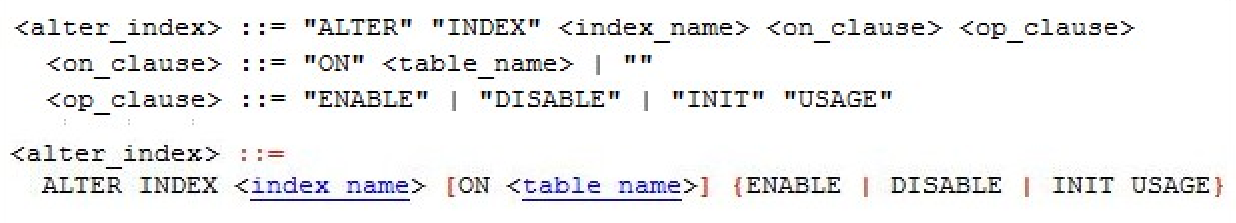
Probably nobody would teach geometry without graphs even if the figures are more precise than the drawing ever can be.

In both cases the usefulness of diagrams is obvious. But does the same consideration apply to other types of diagrams?

EBNF (Extended Backus Naur Form)

BNF (Backus Naur Form) is a meta language to define exactly the grammar of a context-free language. Often parsers are derived from the BNF. Most of the time an extended form (EBNF) is used for this purpose. The metalanguage can easily be adopted. It needs one page of explanation.

The original BNF was extended by additional meta characters. Today EBNF is the standard.



```
<alter_index> ::= "ALTER" "INDEX" <index_name> <on_clause> <op_clause>
  <on_clause> ::= "ON" <table_name> | ""
  <op_clause> ::= "ENABLE" | "DISABLE" | "INIT" "USAGE"

<alter_index> ::=
  ALTER INDEX <index name> [ON <table name>] {ENABLE | DISABLE | INIT USAGE}
```

Fig. 1: BNF as text versus EBNF in HTML

In brief: An EBNF grammar consists of production rules. A rule starts with its name followed by an assignment "::=". The rule consists of clauses: Pure text - terminals, references to other rules - nonterminals, options enclosed in brackets "[]", alternatives separated by bars "|". repetitions indicated by three dots "…" and nameless rules in curly braces "{}". That's all one has to know of the metalanguage used here.

EBNF is also used to document a language. Many times you will find it in the appendix of a handbook. And there it stays unused because it consists of rules in rules. This nesting forces you to thumb through the appendix. At the end you do not know where you came from. This is a big drawback.

Highlighting and Links

Highlighting of metacharacters distinguishes them from language characters, no escapes or quotes are necessary any more. It helps if the grammar is in HTML format with links to rules. Then the thumb is replaced by the mouse. It's faster but not better.

Syntax Diagrams

The transformation of a syntax rule into a diagram, also known as raiload diagrams, improves the comprehension of a BNF notation.

Rectangles are used for constants, hexagons for rule references, optional paths for options and alternatives and backspace paths with arrows for repetitions (not contained in figure 2).

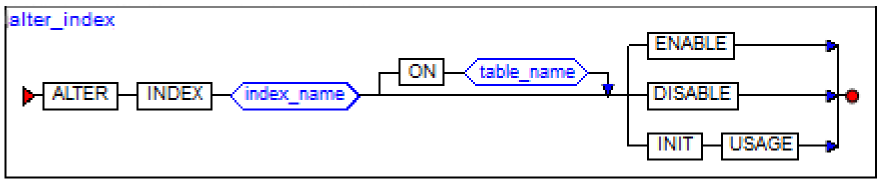


Fig. 2: <alter_index> rule turned into a diagram

It is obviously easier to grasp the format of the diagram rule without losing precision. This is still a static approach. It could be part of a manual. See e.g. Oracle's DB documentation.Drop Down Diagrams

Dynamics come in with the interactive exploded view of the subrules if they become clickable. One might call them drop down diagrams.

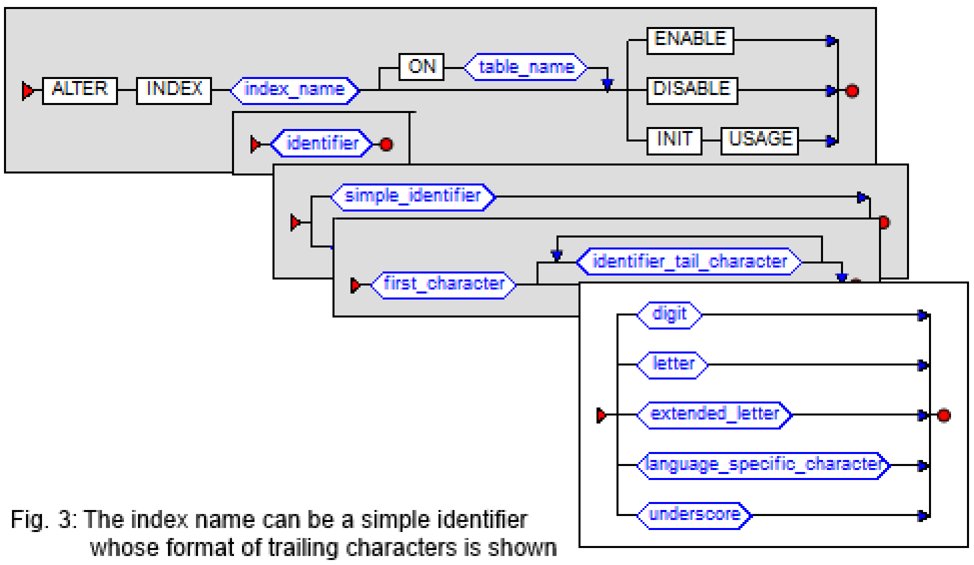


Fig. 3: The index name can be a simple identifier whose format of trailing characters is shown

The rules are visualized in their individual context. This approach could be implemented by using pictures of each rule in a grammar. The clicked subrule is placed directly under the rule reference.

Inner Expansion of Diagrams

The best presentation will be an inline expansion of a rule.

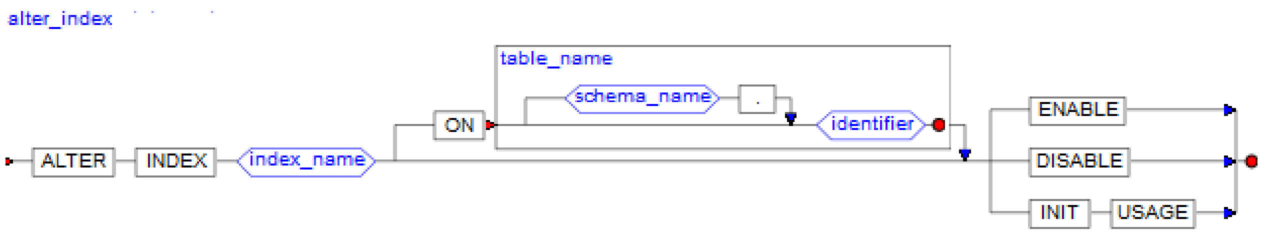


Fig. 4: Diagram in diagram: The <table:name> rule is exploded inline.

But here one stumbles against the limits of conventional HTML. Because of the multitudinous combination of rules the diagrams have to be redrawn. They cannot be prepared beforehand. The tool of choice for inline expansion is SVG.
One can see at a glance how the SQL statement should be written. Of course, nested diagrams can be drawn to arbitrary depth. The limit is the size of the screen. But scrolling can be applied.Interactive Reduction of Complexity

A valid and compilable language statement containes no option or alternatve anymore.

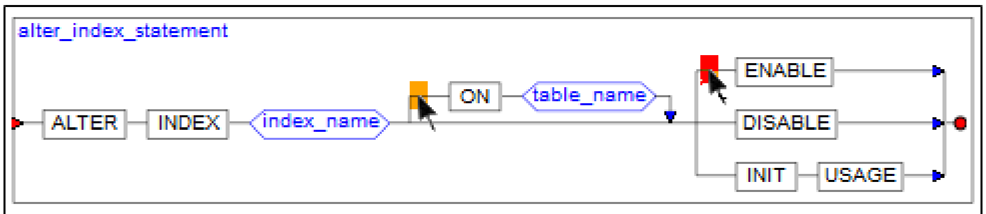


Fig. 5: Selecting an option

In addition to exploding rules, they can be collapsed by selecting alternatives or dropping options. As a reminder of a collapsed clause a colored rectangle will appear. There are hidden rectangles in front of options and alternatives. If the mouse moves over it, they will appear. If clicked, the option or alternative will be selected.

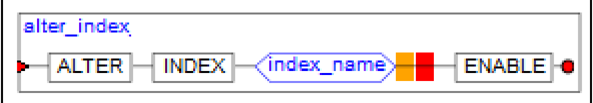


Fig. 6: With dropped option and selected alternative

An option can either be dropped or selected.

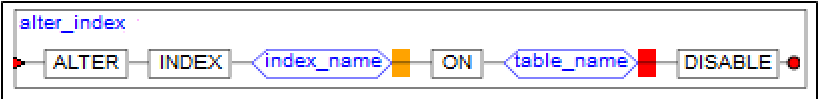


Fig. 7: With selected option and selected alternative

The colored rectangles – ocher for option and red for alternatives - are reminders of the reduction. If clicked, the original will be redrawn.
If only one line of clauses is left, a statement template will be printed, like:

```
ALTER INDEX <index_name> ON <table_name> DISABLE
```

Redundancy Reduction

A BNF and even an EBNF notation can be lengthy.

```
<bool_predicate> ::=
  <column spec> [IS [NOT] TRUE | IS [NOT] FALSE]
```

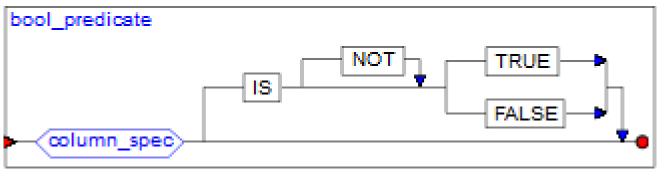


Fig. 8: The diagram is a reduced version of the BNF rule

Therefore an algorithm has been developed to reduce redundancy. Depending on the specified depth rules can be automatically reduced to their lowest term.Discussions

Discussions

Here are the answers to the questions from the beginning:

Is visualization more comprehensive than written text?
Yes

Is it precise?

Yes, it is a one to one representation of the EBNF, in case of the redundency reduction it is even better.

Is it intuitive?

Yes, the diagrams need almost no explanation or legend

Does its interaction provide a better understanding?

Yes, drop down rules and inline expansion of rules avoid thumbing through the rules. Selecting options and/or alternatives reduce complexity.

Is it worthwhile (cost effectiveness)?

If an EBNF language definition of the grammar is available, only a batch process is required. If the EBNF notation is different, the EBNF parser has to be adapted. Error corrections might be required.

Is it more than a toy?

We will see. It depends how many clicks will be generated. A predecessor had some hundred thousand clicks per year with only one grammar

What are the limits?

The interactive diagrams are a reference system for the syntax only. There is no semantic explanation. It addresses experts.

Is the technical base – Javascript, HTML plus embedded SVG - sufficient?

A simple yes

There exists a licencefree database with a „zoo“ of grammars.
Some hundred BNFs on various languages were collected. The intention is to convert them to a standard BNF format, generate the diagrams and make them available to the public.

Diagram Production Gontrol Flow

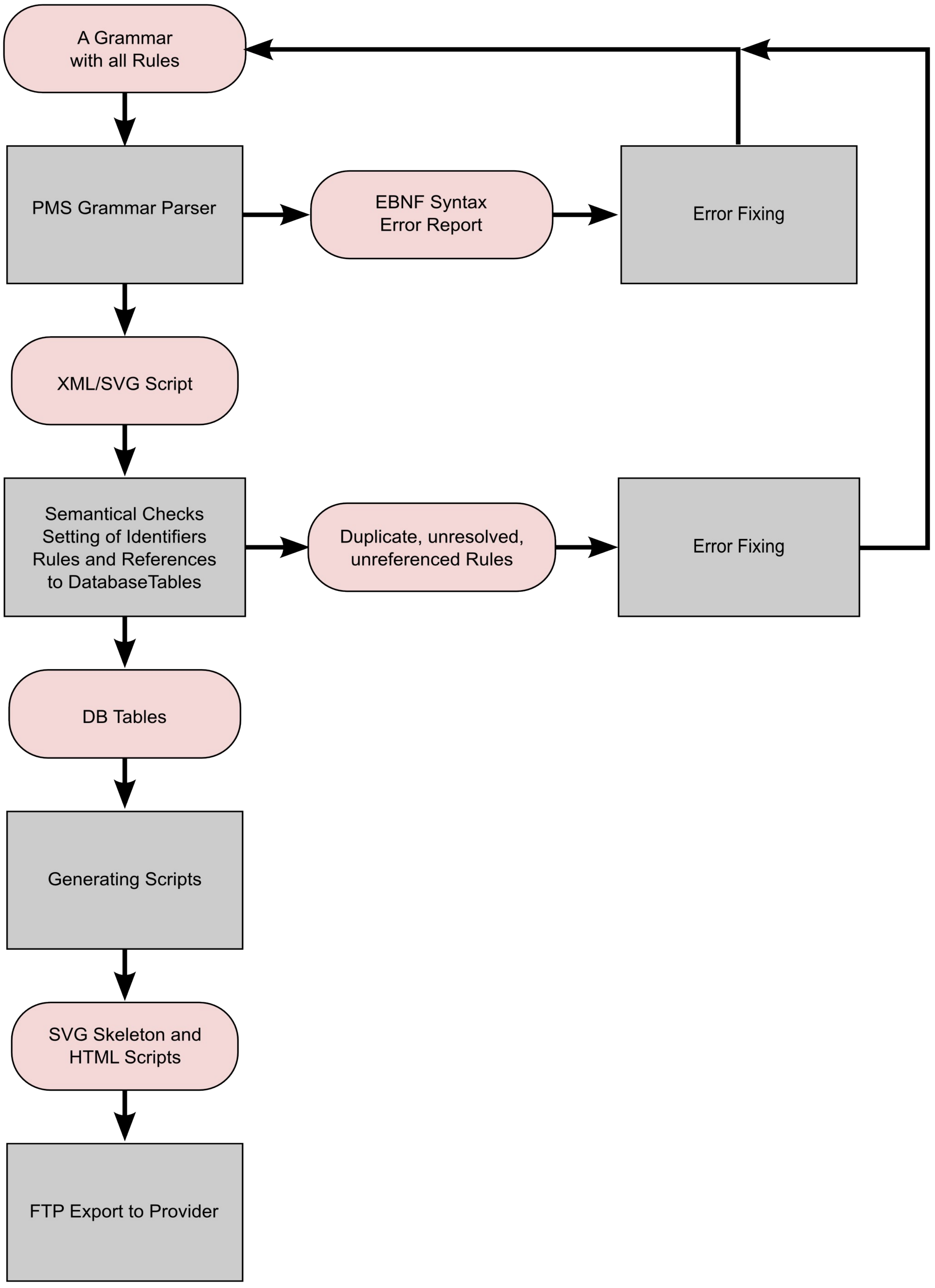


Fig. 9 Production Control Flow

Interactive Syntax Diagrams Application

The application runs with the following browsers:

- Chrome
- Opera
- Safari
- Midori
- Internet Explorer
- Firefox

Java scripting must be enabled.

The application starts with a list of supported grammars. If one grammar is selected, a list of all rules in alphabetical order appears. From here rules can be chosen.

An HTML script with an embedded SVG skeleton will be fetched. The skeleton contains nested <svg> and <desc> nodes with all information required to generate all graphical elements of the diagram. The Javascript functions will draw all these elements.

When a rule is clicked the corresponding SVG skeleton will be fetched and inserted into the original skeleton. The combined skeletons will be redrawn.

A similar process takes place when collapsing options and alternatives. The surplus <svg> nodes will be inactivated in the current skeleton and the diagram is redrawn according to the directives in the skeleton.

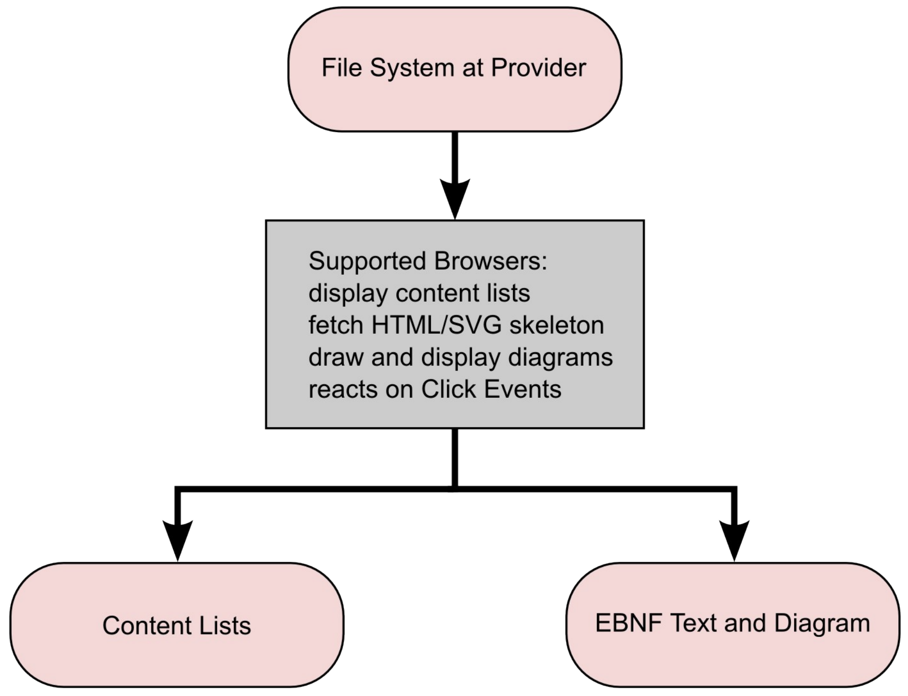
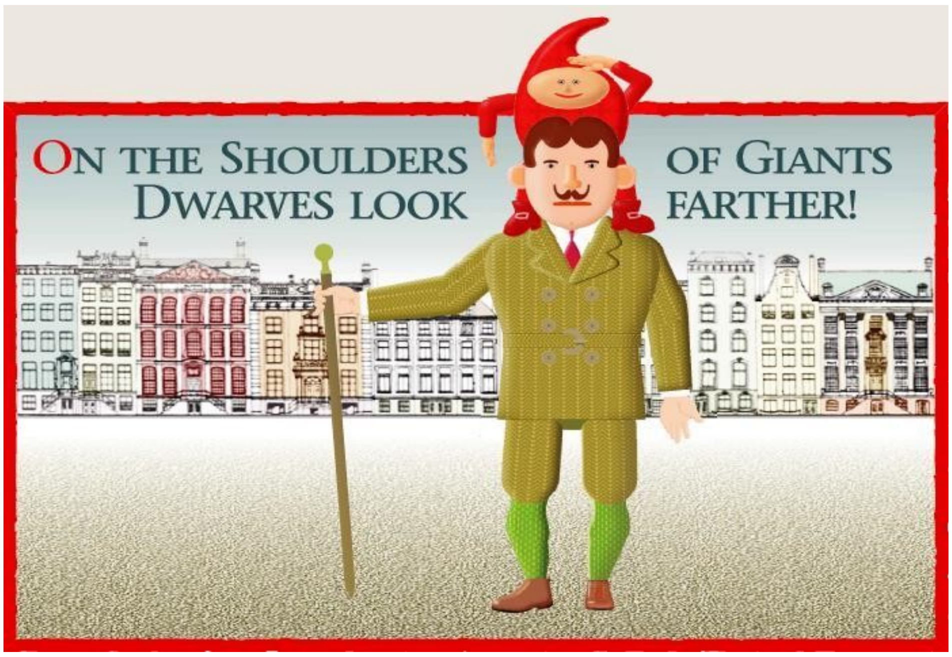


Fig. 10 Web Application of Syntax Diagrams

On the Shoulder of Giants Dwarves look Father

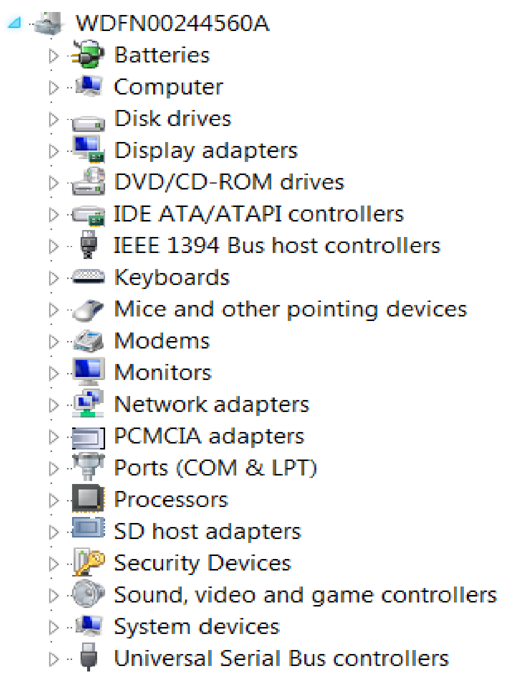


The author started in 1963 in the IT business. He is still astonished about the development in hard- and software. The presented application could not have been written fifty years ago. The hardware would have cost some million Dollar or Euro. The variety of today's software was not available.

This application consists only of couple of thousand lines of code in Perl and Javascript. Some of you could write it in a couple of weeks. This is the dwarf.

The acknowledgment goes to hard- and software developers. There are the giants. The hardware is inexpensive and most of the software used is free:

Typical Hardware of a Personal Computer



```
WDFN00244560A
    Batteries
    Computer
    Disk drives
    Display adapters
    DVD/CD-ROM drives
    IDE ATA/ATAPI controllers
    IEEE 1394 Bus host controllers
    Keyboards
    Mice and other pointing devices
    Modems
    Monitors
    Network adapters
    PCMCIA adapters
    Ports (COM & LPT)
    Processors
    SD host adapters
    Security Devices
    Sound, video and game controllers
    System devices
    Universal Serial Bus controllers
```

Software Used

Internet
Provider with a File System
FTP
2 Browser Debuggers
XML Notepad
Notepad++
LibreOffice
HTML point
HTML
JavaScript + SVG
6 different Browsers with SVG Support
PMS for BNF Parser
C/C++
Perl-IDE
Batch: Perl + XML + SQL
DB-Studio
DB: MaxDB
OS: Windows 7