This is work based on:

TOUCH PERIODIC TABLE of ELEMENTS

HP Prime

By Mickaël Nicotera

(revised by Miguel Toro)

http://www.hpcalc.org/details.php?id=7556

(hosted by Erich Rechlin's HP Calc)

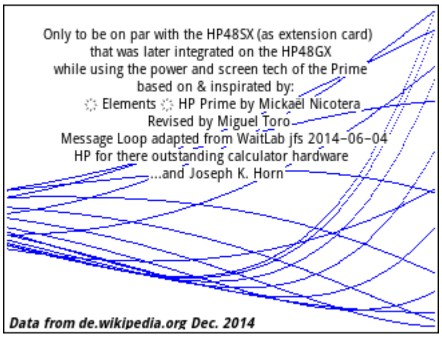
The data was taken from [http://de.wikipedia.org/Periodentafel during December 2014](http://de.wikipedia.org/Periodentafel)

Do not be silly and rely for important things in life on this data. Cross check, double check if you really need to be sure.
I tried to be systematic, but hey a typo is easily made…

Quick :

1.) **If you just want to use the application**
2.) **If you are interested in the new features**
3.) **If you would like to update/improve the application and/or database**

Credit where credit is due:



Only to be on par with the HP48SX (as extension card)
that was later integrated on the HP48GX
while using the power and screen tech of the Prime
based on & inspirated by:
Elements HP Prime by Mickaël Nicotera
Revised by Miguel Toro
Message Loop adapted from WaitLab jfs 2014-06-04
HP for there outstanding calculator hardware
...and Joseph K. Horn

Data from de.wikipedia.org Dec. 2014

Also see on HP Calc:

SkeletonApp: **Event Driven Framework**
empty shell to start from, go right to the important stuff and do not mess with system immanent grunt work
In fact this application is an implementation of

HPLang: **A Notpad++ 'User language' with highlighting code for the HP Prime**
I find the connectivity kit lacks certain invaluable but basic features like 'find' or 'line numbering'

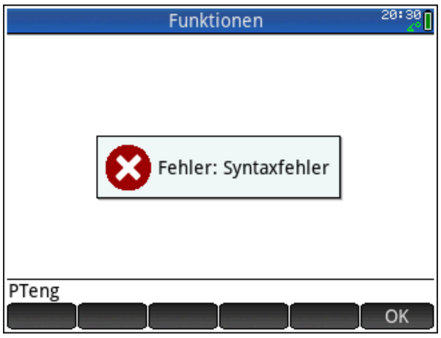
Fonts: **After messing around with trying to figure out the width of a text (GetTextExtent), finally I found that the textout_p function indeed has this built in. It is buried undocumented in the c1 parameter. There is right align, left align, center text, bold, italic, underline, strikethrough, various font sizes and much more.**

1. **Install and use the application**

    1. **Download/Copy the**
        a. **‘PTger (for German language)**
        b. **‘PTeng (for English language)**

    to the Program Catalog and run it.

    CAUTION:
    If you run more than one copy of the same application it will show a syntax error:

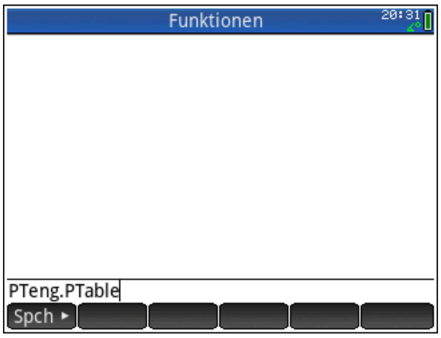


Fehler: Syntaxfehler

PTeng

That is because both export the same entry function PTable
You may run one or the other by using a fully qualified name e.g:

PTeng.PTable
PTger.PTable



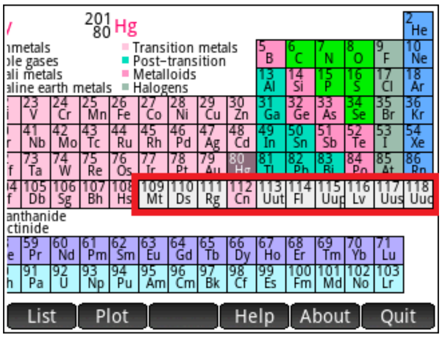
PTeng.PTable

2. Description of the new features

This revision improves on the following areas :

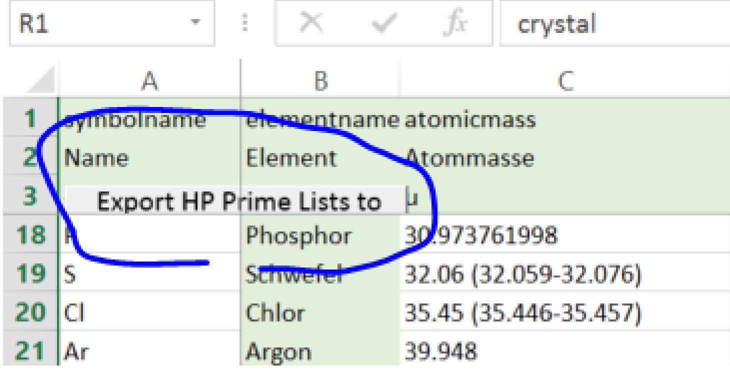
1. Added element 113 to 118
2. Use a ‘database’ to create the ‘data dictionary’ of the application
   Creates ‘automagically’ a new ‘data dictionary’
3. Adds a lot more properties to the database
   Adds a second page of properties to each element
4. Ability to move freely on the table ‘visually’
   by going up and down or left and right
   by touching the element on the screen
   update the overview screen by showing the selected element
5. Adds the ability to ‘scroll’ through the properties
   using up and down keys
   by drag gesture up and down on the screen
6. Adds the ability to ‘scroll’ through the elements on the properties page
   using left and right keys
   by drag gesture left and right on the screen
   selectable properties
7. Adds ability to support multiple languages
8. Add graph of selected properties through own plotting facility
   select the property to plot using touch or keys
   using left and right keys to scroll through the elements
   using touch gesture to select visually the element on the graph
   option to draw the graph as ‘scatter plot’ or with connected lines
9. Add sortable List View
   Sort listview by touching the colums header
   Select single element with touch gesture and press Enter to see its properties

1. Added element 113 to 118



2. Use a ‘database’ to create the ‘data dictionary’ of the application

See developers guide later in this paper how to manipulate the ’database’
Creates ‘automagically’ a new ‘data dictionary’



R1 | crystal

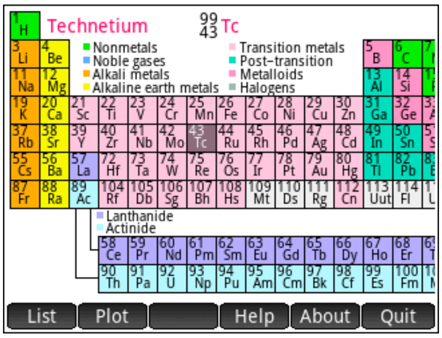
	A	B	C
1	symbolname	elementname	atomicmass
2	Name	Element	Atommasse
3	Export HP Prime Lists to		µ
18	P	Phosphor	30.973761998
19	S	Schwefel	32.06 (32.059-32.076)
20	Cl	Chlor	35.45 (35.446-35.457)
21	Ar	Argon	39.948

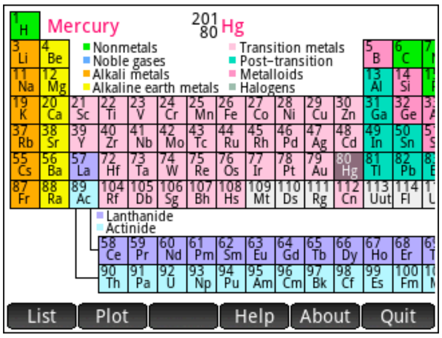
3. **Adds a lot more properties to the database**

There is a total of 32 properties

Name		*(existing)*
Number		*(existing)*
Element		*(existing)*
General		
	Category	*(new)*
	Group, Period, Block	*(new)*
	Look alike	*(new)*
	CAS Number	*(new)*
	Mass on Earth shell	*(new)*
	Discovery	*(existing)*
Atomic		
	Atomic Weight	*(existing)*
	Atomic Radius	*(new)*
	Covalent Radius	*(new)*
	Van-der-Waals-Radius	*(new)*
	Elektron Configuration	*(new)*
	First Ionisation Energy	*(new)*
Physical		
	Phase	*(new)*
	Crystal Structure	*(new)*
	Density	*(existing)*
	Melting Point	*(existing)*
	Boiling Point	*(existing)*
	Atomic Volume	*(new)*
	Heat of Vaporization	*(new)*
	Heat of Fusion	*(new)*
	Heat Capacity	*(new)*
	Electrical Conductivity	*(new)*
	Thermal Conductivity	*(new)*
Chemical		
	Oxidation States	*(new)*
	Oxide	*(new)*
	Elektronegativity	*(existing)*

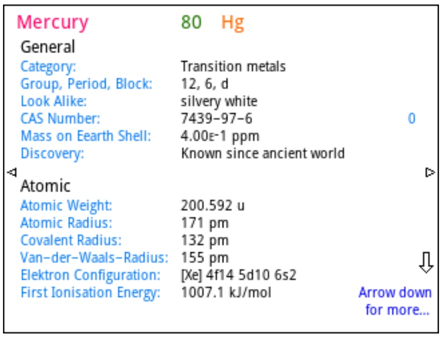
4. **Update the overview screen by showing the selected element**





Show Name of Element selected **Show Name of Element selected**

5. **Adds a second page of properties to each element**



Mercury 80 Hg

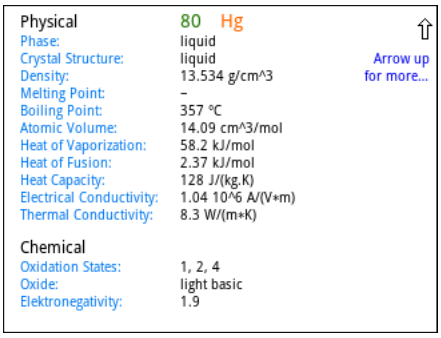
General

Category: Transition metals
Group, Period, Block: 12, 6, d
Look Alike: silvery white
CAS Number: 7439-97-6 0
Mass on Earth Shell: 4.00E-1 ppm
Discovery: Known since ancient world

Atomic

Atomic Weight: 200.592 u
Atomic Radius: 171 pm
Covalent Radius: 132 pm
Van-der-Waals-Radius: 155 pm
Elektron Configuration: [Xe] 4f14 5d10 6s2
First Ionisation Energy: 1007.1 kJ/mol

Arrow down for more...



Physical 80 Hg

Phase: liquid
Crystal Structure: liquid
Density: 13.534 g/cm^3
Melting Point: -
Boiling Point: 357 °C
Atomic Volume: 14.09 cm^3/mol
Heat of Vaporization: 58.2 kJ/mol
Heat of Fusion: 2.37 kJ/mol
Heat Capacity: 128 J/(kg.K)
Electrical Conductivity: 1.04 10^6 A/(V*m)
Thermal Conductivity: 8.3 W/(m*K)

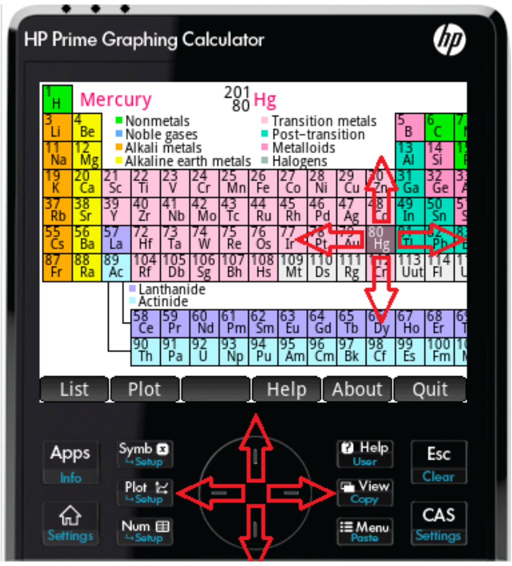
Arrow up for more...

Chemical

Oxidation States: 1, 2, 4
Oxide: light basic
Elektronegativity: 1.9

Page 1 **Page 2**

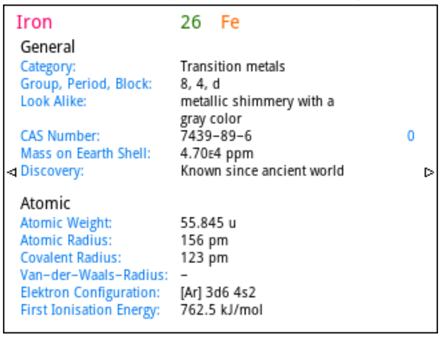
6. **Ability to move freely on the table 'visually'**



Use the arrow keys up, down & left, right for movement around the table. Not necessary by number but by visual appearance. Use the touch gesture on the screen to move freely left and right.

Update the overview screen by showing the selected element if you tap another element.

7. **Adds the ability to 'scroll' through the properties**



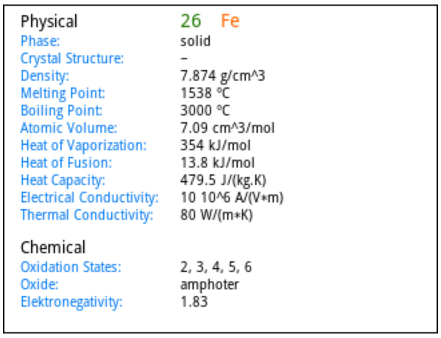
Iron 26 Fe

General

Category: Transition metals
Group, Period, Block: 8, 4, d
Look Alike: metallic shimmery with a gray color
CAS Number: 7439-89-6 0
Mass on Earth Shell: 4.70E4 ppm
Discovery: Known since ancient world

Atomic

Atomic Weight: 55.845 u
Atomic Radius: 156 pm
Covalent Radius: 123 pm
Van-der-Waals-Radius: -
Elektron Configuration: [Ar] 3d6 4s2
First Ionisation Energy: 762.5 kJ/mol



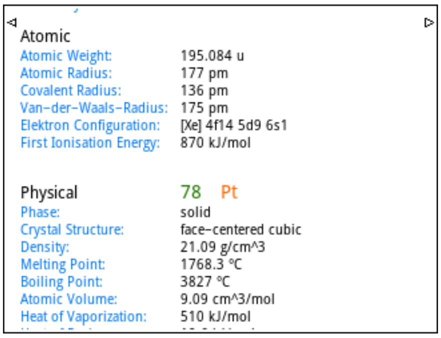
Physical 26 Fe

Phase: solid
Crystal Structure: -
Density: 7.874 g/cm^3
Melting Point: 1538 °C
Boiling Point: 3000 °C
Atomic Volume: 7.09 cm^3/mol
Heat of Vaporization: 354 kJ/mol
Heat of Fusion: 13.8 kJ/mol
Heat Capacity: 479.5 J/(kg.K)
Electrical Conductivity: 10 10^6 A/(V*m)
Thermal Conductivity: 80 W/(m*K)

Chemical

Oxidation States: 2, 3, 4, 5, 6
Oxide: amphoter
Elektronegativity: 1.83

using up and down keys



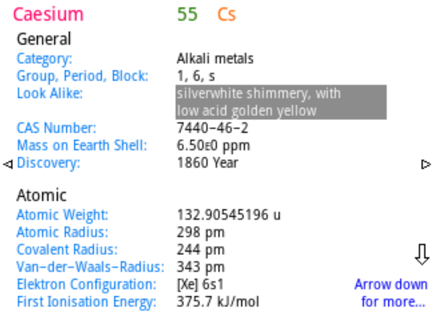
Atomic

Atomic Weight:	195.084 u
Atomic Radius:	177 pm
Covalent Radius:	136 pm
Van-der-Waals-Radius:	175 pm
Elektron Configuration:	[Xe] 4f14 5d9 6s1
First Ionisation Energy:	870 kJ/mol

Physical	78 Pt
Phase:	solid
Crystal Structure:	face-centered cubic
Density:	21.09 g/cm^3
Melting Point:	1768.3 °C
Boiling Point:	3827 °C
Atomic Volume:	9.09 cm^3/mol
Heat of Vaporization:	510 kJ/mol

by drag gesture up and down on the screen

8. Adds the ability to ‘scroll’ through the elements on the properties page
    1. by pressing left or right keys Scrolls on after the other
    2. by drag gesture left and right on the screen Scrolls very fast through elements
    3. There is a feature of individually selectable properties



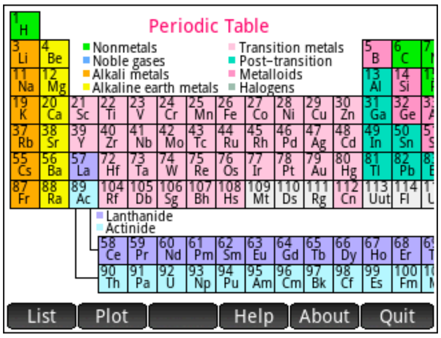
Caesium	55 Cs
General	
Category:	Alkali metals
Group, Period, Block:	1, 6, s
Look Alike:	silverwhite shimmery, with low acid golden yellow
CAS Number:	7440-46-2
Mass on Eearth Shell:	6.50E0 ppm
Discovery:	1860 Year

Atomic	
Atomic Weight:	132.90545196 u
Atomic Radius:	298 pm
Covalent Radius:	244 pm
Van-der-Waals-Radius:	343 pm
Elektron Configuration:	[Xe] 6s1
First Ionisation Energy:	375.7 kJ/mol

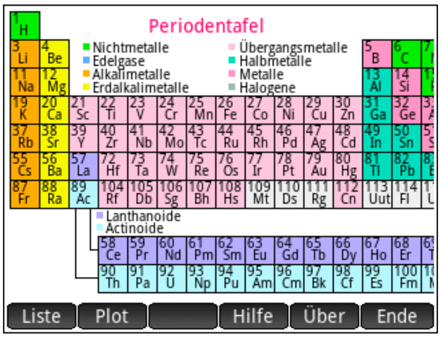
Arrow down for more...

(However no functionality has been implemented, I wanted to push values on the stack but there is none)

9. Adds ability to support multiple languages



PTeng.PTable: English

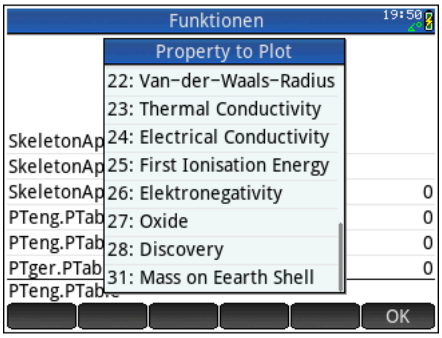


PTger.PTable: German

You may add your own language of you desire. Please read developers section on how to add another language.

10. Add graph of selected properties through own plotting facility

    select the property to plot using touch or keys



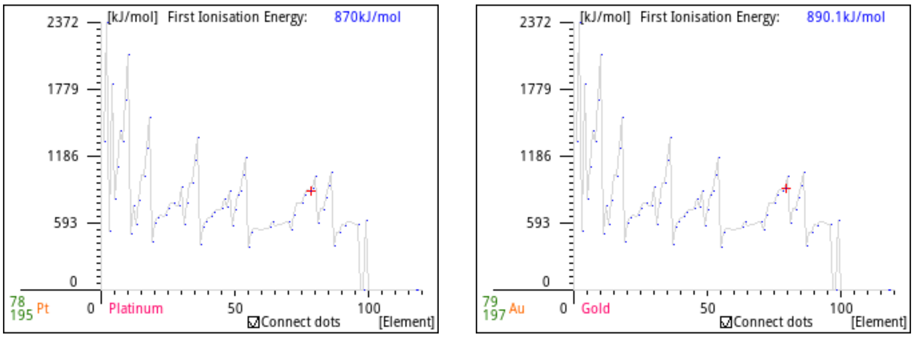
Funktionen 19:50

Property to Plot

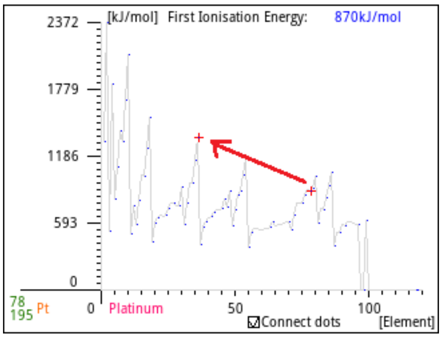
22: Van-der-Waals-Radius
23: Thermal Conductivity
24: Electrical Conductivity
25: First Ionisation Energy
26: Elektronegativity
27: Oxide
28: Discovery
31: Mass on Earth Shell

SkeletonAp
SkeletonAp
SkeletonAp 0
PTeng.PTab 0
PTeng.PTab 0
PTger.PTab 0
PTeng.PTable

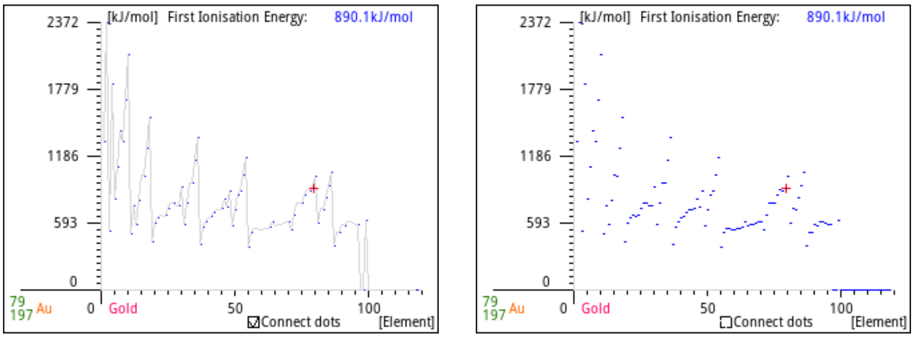
OK



using left and right keys to scroll through the elements

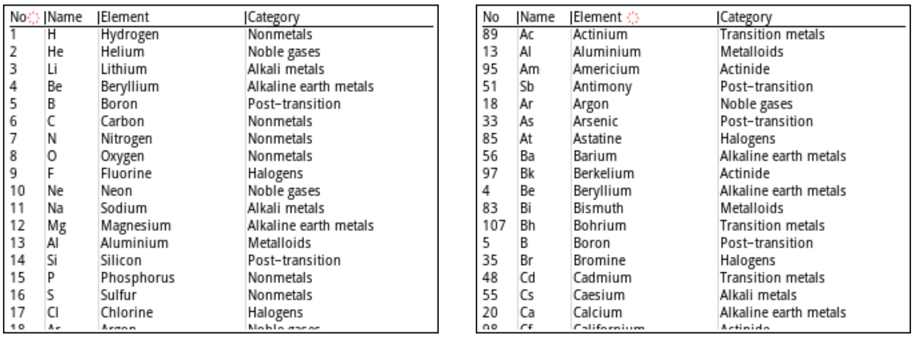


using touch gesture to select visually the element on the graph



option to draw the graph as 'scatter plot' or with connected lines

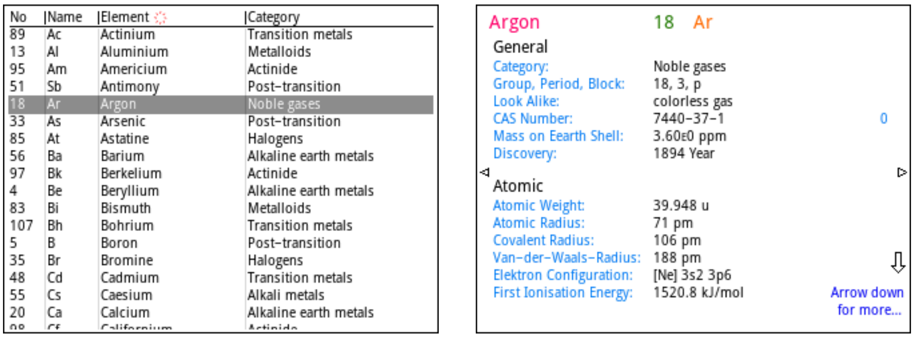
11. Add sortable List View



No	Name	Element	Category
1	H	Hydrogen	Nonmetals
2	He	Helium	Noble gases
3	Li	Lithium	Alkali metals
4	Be	Beryllium	Alkaline earth metals
5	B	Boron	Post-transition
6	C	Carbon	Nonmetals
7	N	Nitrogen	Nonmetals
8	O	Oxygen	Nonmetals
9	F	Fluorine	Halogens
10	Ne	Neon	Noble gases
11	Na	Sodium	Alkali metals
12	Mg	Magnesium	Alkaline earth metals
13	Al	Aluminium	Metalloids
14	Si	Silicon	Post-transition
15	P	Phosphorus	Nonmetals
16	S	Sulfur	Nonmetals
17	Cl	Chlorine	Halogens
18	Ar	Argon	Noble gases

No	Name	Element	Category
89	Ac	Actinium	Transition metals
13	Al	Aluminium	Metalloids
95	Am	Americium	Actinide
51	Sb	Antimony	Post-transition
18	Ar	Argon	Noble gases
33	As	Arsenic	Post-transition
85	At	Astatine	Halogens
56	Ba	Barium	Alkaline earth metals
97	Bk	Berkelium	Actinide
4	Be	Beryllium	Alkaline earth metals
83	Bi	Bismuth	Metalloids
107	Bh	Bohrium	Transition metals
5	B	Boron	Post-transition
35	Br	Bromine	Halogens
48	Cd	Cadmium	Transition metals
55	Cs	Caesium	Alkali metals
20	Ca	Calcium	Alkaline earth metals
98	Cf	Californium	Actinide

Sort listview by touching the colums head
(Notice the red circle next to the columns head)



No	Name	Element	Category
89	Ac	Actinium	Transition metals
13	Al	Aluminium	Metalloids
95	Am	Americium	Actinide
51	Sb	Antimony	Post-transition
18	Ar	Argon	Noble gases
33	As	Arsenic	Post-transition
85	At	Astatine	Halogens
56	Ba	Barium	Alkaline earth metals
97	Bk	Berkelium	Actinide
4	Be	Beryllium	Alkaline earth metals
83	Bi	Bismuth	Metalloids
107	Bh	Bohrium	Transition metals
5	B	Boron	Post-transition
35	Br	Bromine	Halogens
48	Cd	Cadmium	Transition metals
55	Cs	Caesium	Alkali metals
20	Ca	Calcium	Alkaline earth metals
98	Cf	Californium	Actinide

Argon 18 Ar

General

Category:	Noble gases
Group, Period, Block:	18, 3, p
Look Alike:	colorless gas
CAS Number:	7440-37-1
Mass on Eearth Shell:	3.60E0 ppm
Discovery:	1894 Year

Atomic

Atomic Weight:	39.948 u
Atomic Radius:	71 pm
Covalent Radius:	106 pm
Van-der-Waals-Radius:	188 pm
Elektron Configuration:	[Ne] 3s2 3p6
First Ionisation Energy:	1520.8 kJ/mol

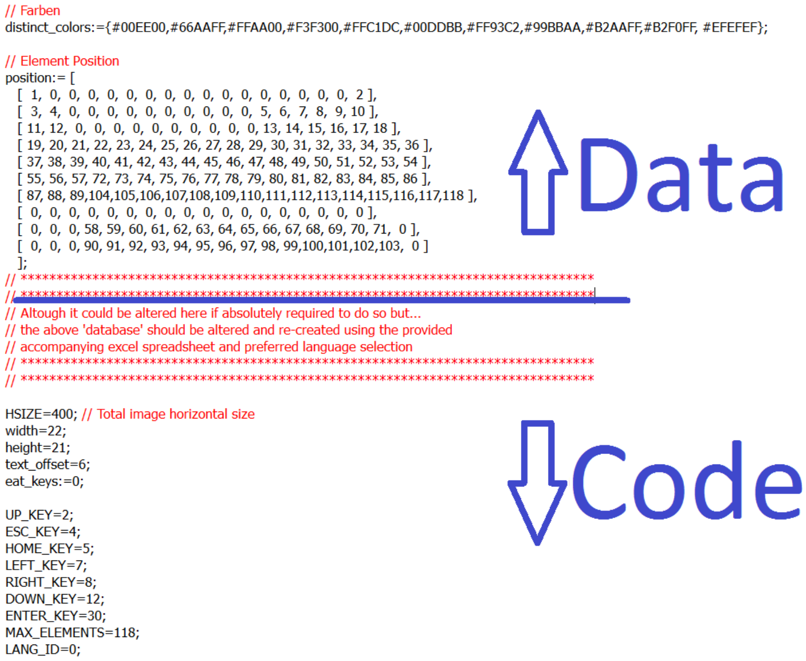
Select single element with touch gesture and press Enter to see its properties

If you would like to update/improve the application and/or database

3. **Update/improve the application and/or database**

   Quick rundown:

   1. **Update Excel sheet(s) 'PTable' and optionally Language**
   2. **Create a new 'data database' by exporting/converting to 'HPPL' (HP Prime Language')**
   3. **Update the first part (until the clearly marked comments) of the applications code through for example Connectivity Kit or Notepad using copy/paste**



```
// Farben
distinct_colors:={#00EE00,#66AAFF,#FFAA00,#F3F300,#FFC1DC,#00DDBB,#FF93C2,#99BBAA,#B2AAFF,#B2F0FF, #EFEFEF};

// Element Position
position:= [
  [ 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2 ],
  [ 3, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 6, 7, 8, 9, 10 ],
  [ 11, 12, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 13, 14, 15, 16, 17, 18 ],
  [ 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36 ],
  [ 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54 ],
  [ 55, 56, 57, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86 ],
  [ 87, 88, 89,104,105,106,107,108,109,110,111,112,113,114,115,116,117,118 ],
  [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ],
  [ 0, 0, 0, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 0 ],
  [ 0, 0, 0, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99,100,101,102,103, 0 ]
  ];
// ****************************************************************************
// ****************************************************************************
// Altough it could be altered here if absolutely required to do so but...
// the above 'database' should be altered and re-created using the provided
// accompanying excel spreadsheet and preferred language selection
// ****************************************************************************
// ****************************************************************************

HSIZE=400; // Total image horizontal size
width=22;
height=21;
text_offset=6;
eat_keys:=0;

UP_KEY=2;
ESC_KEY=4;
HOME_KEY=5;
LEFT_KEY=7;
RIGHT_KEY=8;
DOWN_KEY=12;
ENTER_KEY=30;
MAX_ELEMENTS=118;
LANG_ID=0;
```

   4. **Download (or copy/paste) the new code to the Virtual Calculator or to the real Prime.**
   5. **Run and enjoy.**

Developers Section

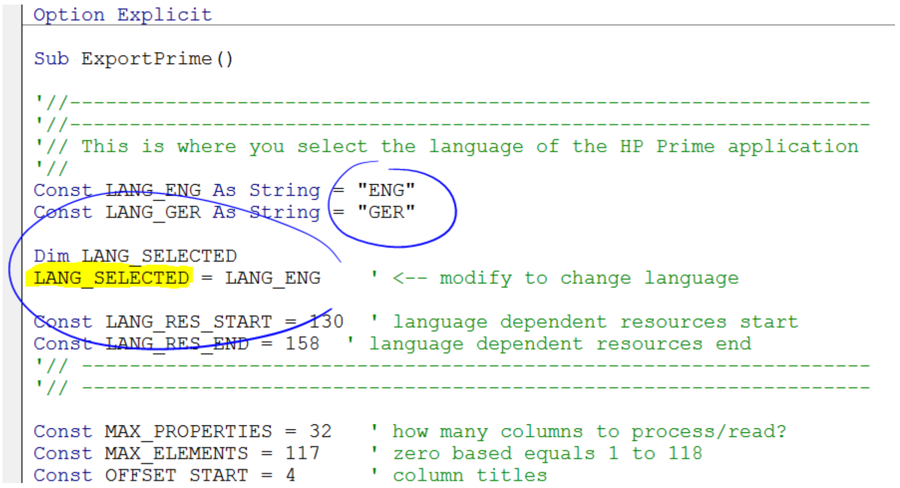
1. **Use a 'database' to create the 'data dictionary' of the application**

The database is created using an Excel Spreadsheet (included) and a Script (included) to create HP Prime compatible language structures you simply copy&paste into the source code.

There are some concepts in using the excel spreadsheet in order to understand it functionality:
Basically when you would like to update a property of an element you need to know which number or text you need to modify. However there is no easy way to figure out in long concatenated lists which one is for example property density on element number 40 Technetium.

Therefore for such a large number of properties and elements it is imperative to use proper mechanism to update.

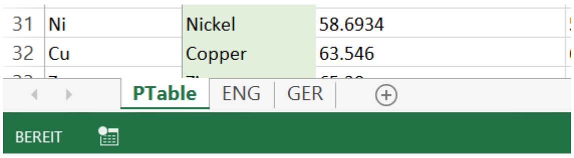
1. **Open the file PTable.xlsm in Excel**
2. **You need to allow scripts to run (script is attached as vba source code below)**
3. **Open Visual Basic for Applications**



```
Option Explicit

Sub ExportPrime()

'//----------------------------------------------------------------------
'//----------------------------------------------------------------------
'// This is where you select the language of the HP Prime application
'//
Const LANG_ENG As String = "ENG"
Const LANG_GER As String = "GER"

Dim LANG_SELECTED
LANG_SELECTED = LANG_ENG     ' <-- modify to change language

Const LANG_RES_START = 130   ' language dependent resources start
Const LANG_RES_END = 158  ' language dependent resources end
'// ---------------------------------------------------------------------
'// ---------------------------------------------------------------------

Const MAX_PROPERTIES = 32    ' how many columns to process/read?
Const MAX_ELEMENTS = 117     ' zero based equals 1 to 118
Const OFFSET_START = 4       ' column titles
```

Selected the language you would like to 'compile' the data dictionary from

NOTE : Currently only English 'ENG' and German 'GER' is supported



31	Ni	Nickel	58.6934
32	Cu	Copper	63.546

You may add your own language(s) by simply adding

a. **a new Worksheet named for example 'ITA' for Italian or 'FRA' for French.**

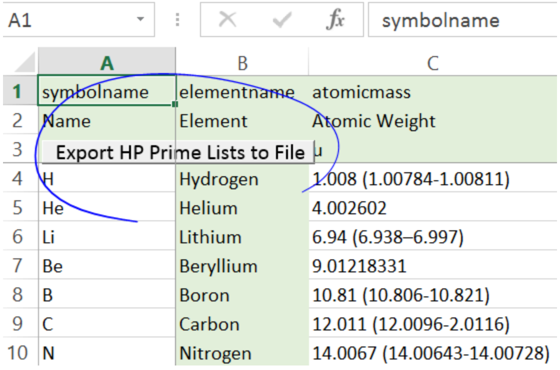
b. **A new**

```
Const LANG_ITA As String = „ITA"

Const LANG_FRA As String = „FRA"
```

c. **And point the currently selected language to :**

```
LANG_SELECTED = LANG_ITA
```

4. **After you selected the language you may create a test run**
By pushing the button 'Export HP Prime Lists to File' it will create a 'new' database



A1 | fx | symbolname

	A	B	C
1	symbolname	elementname	atomicmass
2	Name	Element	Atomic Weight
3			
4	H	Hydrogen	1.008 (1.00784-1.00811)
5	He	Helium	4.002602
6	Li	Lithium	6.94 (6.938–6.997)
7	Be	Beryllium	9.01218331
8	B	Boron	10.81 (10.806-10.821)
9	C	Carbon	12.011 (12.0096-2.0116)
10	N	Nitrogen	14.0067 (14.00643-14.00728)

5. **The concept(s) behind are the following:**

Generally:

- **Light green cells are automatically generated or updated by the script and then exported to the HP Prime compatible file**
  DO NOT modify. (it twill not harm but you may loose your work)
- **White cells are filled by values which are directly exported to HP Prime compatible file.**
  YOU MAY modify as you see fit.

a. **PTable**
One main table consisting of all numerical (language independent) properties

b. **ENG**

English translation of all language dependent properties
As well as the language resources used inside the application itself

c. GER
German translation of all language dependent properties
As well as the language resources used inside the application itself

d. <None>
<Define> translation to be used

PTable:

Static: By Value Values which will be used

Dynamic: By Address Values will be dynamically created using data on this table itself or on the language tables

a. Static : Column 1 through Column 32 (except the six mentioned below) are used to create the properties lists

b. Dynamic : Column 2, 8, 17, 18, 27, 32 are beeing filled by language dependent table

c. Dynamic: Column 34 to 38 are used to create distinct values (reduce footpint)

d. Dynamic: Column 40 to 44 are used to create indices to language dependent strings

e. Dynamic: Column 2
Row 130 to Row 158 : Language dependent application resources strings

Row 1

Static : By Value Universally used 'list identifiers' in the application

Row 2, 3

Dynamic By Address will be updated with values from the language dependent table

ENG, GER, <None>:

Static: By Value Values which will be used by the PTable
If language was selected by the script

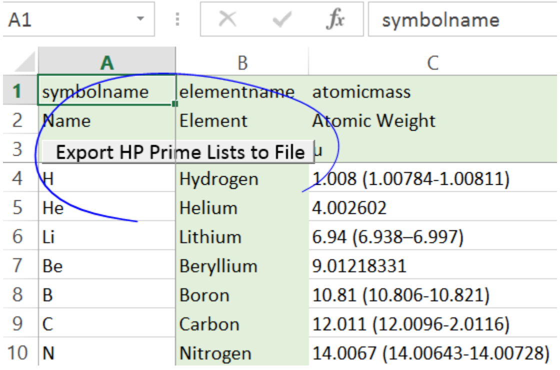
a. Static : Column 2, 8, 17, 18, 27, 32 are being copied to the PTable table to be used to create the properties lists

b.

c. Static: Column 2
Row 130 to Row 158 : Language dependent application resources strings are being copied to the PTable table to be used to create the properties lists

2. Creates ‘automagically’ a new ‘data dictionary’

By pushing the button ‘Export HP Prime Lists to File’ it will create a ‘new’ database

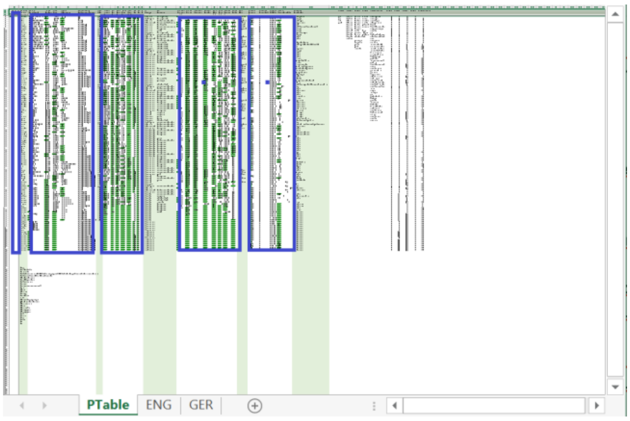


A1 | symbolname

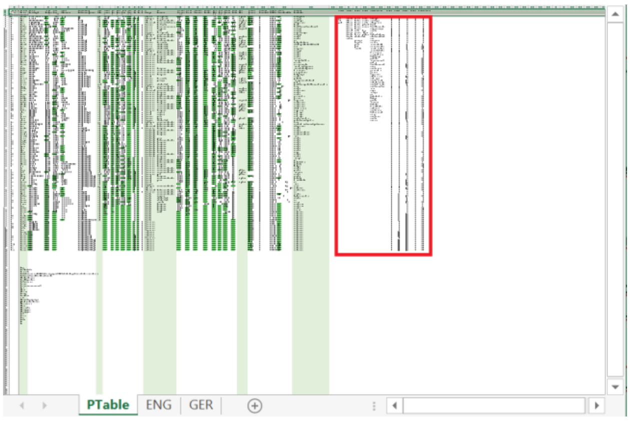
	A	B	C
1	symbolname	elementname	atomicmass
2	Name	Element	Atomic Weight
3	Export HP Prime Lists to File		u
4	H	Hydrogen	1.008 (1.00784-1.00811)
5	He	Helium	4.002602
6	Li	Lithium	6.94 (6.938–6.997)
7	Be	Beryllium	9.01218331
8	B	Boron	10.81 (10.806-10.821)
9	C	Carbon	12.011 (12.0096-2.0116)
10	N	Nitrogen	14.0067 (14.00643-14.00728)

Shows graphically the process of creating resulting table to be exported when hiting button:

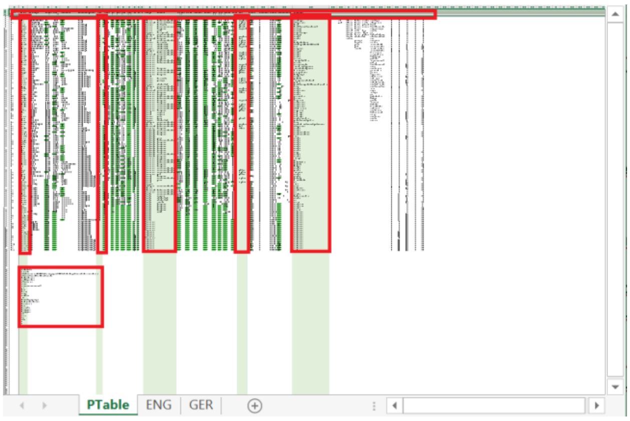
Static portion of table..



...will create the dynamic portion to the right

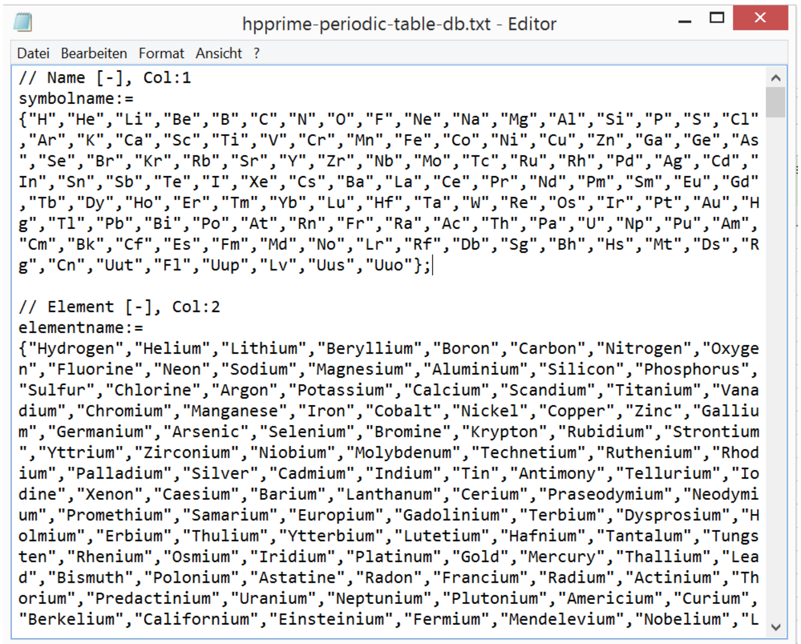


And then copies from the Language table ‘ENG’, ‘GER’, or <None>



Which will eventually result in this combined resulting table

This data is then written to HP Prime compatible source code
It should come up with notepad editor and the source code ready to be used (copied to the application)


hpprime-periodic-table-db.txt - Editor
Datei Bearbeiten Format Ansicht ?

```
// Name [-], Col:1
symbolname:=
{"H","He","Li","Be","B","C","N","O","F","Ne","Na","Mg","Al","Si","P","S","Cl"
,"Ar","K","Ca","Sc","Ti","V","Cr","Mn","Fe","Co","Ni","Cu","Zn","Ga","Ge","As
","Se","Br","Kr","Rb","Sr","Y","Zr","Nb","Mo","Tc","Ru","Rh","Pd","Ag","Cd","
In","Sn","Sb","Te","I","Xe","Cs","Ba","La","Ce","Pr","Nd","Pm","Sm","Eu","Gd"
,"Tb","Dy","Ho","Er","Tm","Yb","Lu","Hf","Ta","W","Re","Os","Ir","Pt","Au","H
g","Tl","Pb","Bi","Po","At","Rn","Fr","Ra","Ac","Th","Pa","U","Np","Pu","Am",
"Cm","Bk","Cf","Es","Fm","Md","No","Lr","Rf","Db","Sg","Bh","Hs","Mt","Ds","R
g","Cn","Uut","Fl","Uup","Lv","Uus","Uuo"};

// Element [-], Col:2
elementname:=
{"Hydrogen","Helium","Lithium","Beryllium","Boron","Carbon","Nitrogen","Oxyge
n","Fluorine","Neon","Sodium","Magnesium","Aluminium","Silicon","Phosphorus",
"Sulfur","Chlorine","Argon","Potassium","Calcium","Scandium","Titanium","Vana
dium","Chromium","Manganese","Iron","Cobalt","Nickel","Copper","Zinc","Galliu
m","Germanium","Arsenic","Selenium","Bromine","Krypton","Rubidium","Strontium
","Yttrium","Zirconium","Niobium","Molybdenum","Technetium","Ruthenium","Rhod
ium","Palladium","Silver","Cadmium","Indium","Tin","Antimony","Tellurium","Io
dine","Xenon","Caesium","Barium","Lanthanum","Cerium","Praseodymium","Neodymi
um","Promethium","Samarium","Europium","Gadolinium","Terbium","Dysprosium","H
olmium","Erbium","Thulium","Ytterbium","Lutetium","Hafnium","Tantalum","Tungs
ten","Rhenium","Osmium","Iridium","Platinum","Gold","Mercury","Thallium","Lea
d","Bismuth","Polonium","Astatine","Radon","Francium","Radium","Actinium","Th
orium","Predactinium","Uranium","Neptunium","Plutonium","Americium","Curium",
"Berkelium","Californium","Einsteinium","Fermium","Mendelevium","Nobelium","L
```

Visual Basic Script for Excel to create a data dictionary for use with PTable
will export HPPRGM compatible source code (used on Hewlett Packard Prime Calculator)

```
Option Explicit

Sub ExportPrime()

'//----------------------------------------------------------------------
'//----------------------------------------------------------------------
'// This is where you select the language of the HP Prime application
'//
Const LANG_ENG As String = "ENG"
Const LANG_GER As String = "GER"

Dim LANG_SELECTED
LANG_SELECTED = LANG_GER    ' <-- modify to change language

Const LANG_RES_START = 130  ' language dependent resources start
Const LANG_RES_END = 166  ' language dependent resources end
'// ---------------------------------------------------------------------
'// ---------------------------------------------------------------------

Const MAX_PROPERTIES = 32   ' how many columns to process/read?
Const MAX_ELEMENTS = 117    ' zero based equals 1 to 118
Const OFFSET_START = 4      ' column titles

Const ROW_08_HEAD = "H4:H121"
Const ROW_17_HEAD = "Q4:Q121"
Const ROW_18_HEAD = "R4:R121"
Const ROW_27_HEAD = "AA4:AA121"
Const ROW_32_HEAD = "AF4:AF121"

Const ROW_34_DATA = "AH4:AH121"
Const ROW_35_DATA = "AI4:AI121"
Const ROW_36_DATA = "AJ4:AJ121"
Const ROW_37_DATA = "AK4:AK121"
Const ROW_38_DATA = "AL4:AL121"

Const ROW_34_START = "AH1:AH121"
Const ROW_35_START = "AI1:AI121"
Const ROW_36_START = "AJ1:AJ121"
Const ROW_37_START = "AK1:AK121"
Const ROW_38_START = "AL1:AL121"

Const ROW_40_START = "AO1:AO121"
Const ROW_41_START = "AP1:AP121"
Const ROW_42_START = "AQ1:AQ121"
Const ROW_43_START = "AR1:AR121"

Dim wksheet As Excel.Worksheet
Dim i               As Integer
Dim j               As Integer
Dim val             As String
Dim title           As String
Dim list            As String
Dim distinctlist08  As String
Dim distinctlist17  As String
Dim distinctlist18  As String
Dim distinctlist27  As String
Dim distinctlist32  As String
Dim unitlist        As String
Dim headerlist      As String
Dim langreslist     As String

Dim c               As Range
Dim n               As Range
Dim p               As Integer

Dim filenum         As Integer
Dim filename        As String
Dim path            As String
Dim ingnorefirst    As Boolean
Dim RetVal          As Integer


Dim LIGHT_GREEN As Long
LIGHT_GREEN = RGB(226, 239, 218)



Set wksheet = Application.Workbooks(1).Worksheets("PTable")

'// first copy the applicable rows from the language dependent sheet to the PTable sheet
'// (this will also copy the translated strings embedded into the application
For i = 0 To 32
    Select Case i
        Case 2, 8, 17, 18, 27, 32
            Application.Workbooks(1).Sheets(LANG_SELECTED).Columns(i).Copy Destination:=wksheet.Columns(i)
            wksheet.Columns(i).Interior.Color = LIGHT_GREEN
    End Select
Next i
For i = 2 To 3
    '// second copy applicable language dependent labels row from language dependent sheet
    '// third copy applicable unis dependent labels row from language dependent sheet
    Application.Workbooks(1).Sheets(LANG_SELECTED).Rows(i).Copy Destination:=wksheet.Rows(i)
    wksheet.Rows(i).Interior.Color = LIGHT_GREEN
Next i

' Debug.Print wksheet.Name

filename = "hpprime-periodic-table-db.txt"
path = Application.ActiveWorkbook.path
filenum = FreeFile
Open path & "\" & filename For Output As #filenum
Print #filenum, "#pragma mode( separator(.,;) integer(h64) )"

'// empty row 40 first
wksheet.Range(ROW_40_START).Clear
'// empty row 40 first
wksheet.Range(ROW_41_START).Clear
'// empty row 40 first
wksheet.Range(ROW_42_START).Clear
'// empty row 40 first
wksheet.Range(ROW_43_START).Clear

'// empty row 34 first
wksheet.Range(ROW_34_START).Clear
'// empty row 35 first
wksheet.Range(ROW_35_START).Clear
'// empty row 36 first
wksheet.Range(ROW_36_START).Clear
'// empty row 37 first
wksheet.Range(ROW_37_START).Clear
'// empty row 38 first
wksheet.Range(ROW_38_START).Clear

'// for column 8 we want distinct values pasted into row 34
'// note that we start at 3, not 4 as one would expect
'// AdvancedFilter assumes the first entry as the columns header
wksheet.Range(ROW_08_HEAD).AdvancedFilter Action:=xlFilterCopy, CopyToRange:=ActiveSheet.Range("AH4"),
Unique:=True
'// copy the title name and prepend it with "distinct_"
wksheet.Cells(1, 34).Value = "distinct_" & wksheet.Cells(1, 8).Value
'// get the newly created title and begin list
distinctlist08 = wksheet.Cells(1, 34).Value & ":={"
'// loop through and build up the list
ingnorefirst = False
For Each c In wksheet.Range(ROW_34_DATA)
    If c.Value <> "" And ingnorefirst = True Then
        distinctlist08 = distinctlist08 & Chr(34) & c.Value & Chr(34) & ","
    End If
    ingnorefirst = True
Next c
'// strip last superflous comma and close list
distinctlist08 = Left(distinctlist08, Len(distinctlist08) - 1) & "};"
'// once we have the distinct list we can replace the value with an index pointing to the distinct_list
wksheet.Cells(1, 40).Value = "index_" & wksheet.Cells(1, 8).Value
For Each c In wksheet.Range(ROW_08_HEAD)
    p = -1
    For Each n In wksheet.Range(ROW_34_DATA)
        p = p + 1 ' index in list
        If c.Value = n.Value Then
            wksheet.Cells(c.Row, 40).Value = p
        End If
    Next n
Next c


'// for column 17 we want distinct values pasted into row 35
wksheet.Range(ROW_17_HEAD).AdvancedFilter Action:=xlFilterCopy, CopyToRange:=ActiveSheet.Range("AI4"),
Unique:=True
'// copy the title name and prepend it with "distinct_"
wksheet.Cells(1, 35).Value = "distinct_" & wksheet.Cells(1, 17).Value
'// get the newly created title and begin list
distinctlist17 = wksheet.Cells(1, 35).Value & ":={"
'// loop through and build up the list
ingnorefirst = False
For Each c In wksheet.Range(ROW_35_DATA)
    If c.Value <> "" And ingnorefirst = True Then
        distinctlist17 = distinctlist17 & Chr(34) & c.Value & Chr(34) & ","
    End If
    ingnorefirst = True
Next c
'// strip last superflous comma and close list
distinctlist17 = Left(distinctlist17, Len(distinctlist17) - 1) & "};"
'// once we have the distinct list we can replace the value with an index pointing to the distinct_list
wksheet.Cells(1, 41).Value = "index_" & wksheet.Cells(1, 17).Value
For Each c In wksheet.Range(ROW_17_HEAD)
    p = -1
    For Each n In wksheet.Range(ROW_35_DATA)
        p = p + 1 ' index in list
        If c.Value = n.Value Then
            wksheet.Cells(c.Row, 41).Value = p
        End If
    Next n
Next c

'// for column 18 we want distinct values pasted into row 36
wksheet.Range(ROW_18_HEAD).AdvancedFilter Action:=xlFilterCopy, CopyToRange:=ActiveSheet.Range("AJ4"),
Unique:=True
'// copy the title name and prepend it with "distinct_"
wksheet.Cells(1, 36).Value = "distinct_" & wksheet.Cells(1, 18).Value
'// get the newly created title and begin list
distinctlist18 = wksheet.Cells(1, 36).Value & ":={"
'// loop through and build up the list
ingnorefirst = False
For Each c In wksheet.Range(ROW_36_DATA)
    If c.Value <> "" And ingnorefirst = True Then
        distinctlist18 = distinctlist18 & Chr(34) & c.Value & Chr(34) & ","
    End If
    ingnorefirst = True
Next c
'// strip last superflous comma and close list
distinctlist18 = Left(distinctlist18, Len(distinctlist18) - 1) & "};"
'// once we have the distinct list we can replace the value with an index pointing to the distinct_list
wksheet.Cells(1, 42).Value = "index_" & wksheet.Cells(1, 18).Value
For Each c In wksheet.Range(ROW_18_HEAD)
    p = -1
    For Each n In wksheet.Range(ROW_36_DATA)
        p = p + 1 ' index in list
        If c.Value = n.Value Then
            wksheet.Cells(c.Row, 42).Value = p
        End If
    Next n
Next c

'// for column 27 we want distinct values pasted into row 37
wksheet.Range(ROW_27_HEAD).AdvancedFilter Action:=xlFilterCopy, CopyToRange:=ActiveSheet.Range("AK4"),
Unique:=True
'// copy the title name and prepend it with "distinct_"
wksheet.Cells(1, 37).Value = "distinct_" & wksheet.Cells(1, 27).Value
'// get the newly created title and begin list
distinctlist27 = wksheet.Cells(1, 37).Value & ":={"
'// loop through and build up the list
ingnorefirst = False
For Each c In wksheet.Range(ROW_37_DATA)
    If c.Value <> "" And ingnorefirst = True Then
        distinctlist27 = distinctlist27 & Chr(34) & c.Value & Chr(34) & ","
    End If
    ingnorefirst = True
Next c
'// strip last superflous comma and close list
distinctlist27 = Left(distinctlist27, Len(distinctlist27) - 1) & "};"
'// once we have the distinct list we can replace the value with an index pointing to the distinct_list
wksheet.Cells(1, 43).Value = "index_" & wksheet.Cells(1, 27).Value
For Each c In wksheet.Range(ROW_27_HEAD)
    p = -1
    For Each n In wksheet.Range(ROW_37_DATA)
        p = p + 1 ' index in list
        If c.Value = n.Value Then
            wksheet.Cells(c.Row, 43).Value = p
        End If
    Next n
Next c

'// for column 32 we want distinct values pasted into row 38
wksheet.Range(ROW_32_HEAD).AdvancedFilter Action:=xlFilterCopy, CopyToRange:=ActiveSheet.Range("AL4"),
Unique:=True
'// copy the title name and prepend it with "distinct_"
wksheet.Cells(1, 38).Value = "distinct_" & wksheet.Cells(1, 32).Value
'// get the newly created title and begin list
```

```
distinctlist32 = wksheet.Cells(1, 38).Value & ":={"
'// loop through and build up the list
ingnorefirst = False
For Each c In wksheet.Range(ROW_38_DATA)
    If c.Value <> "" And ingnorefirst = True Then
        distinctlist32 = distinctlist32 & Chr(34) & c.Value & Chr(34) & ","
    End If
    ingnorefirst = True
Next c
'// strip last superflous comma and close list
distinctlist32 = Left(distinctlist32, Len(distinctlist32) - 1) & "};"
'// once we have the distinct list we can replace the value with an index pointing to the distinct_list
wksheet.Cells(1, 44).Value = "index_" & wksheet.Cells(1, 32).Value
For Each c In wksheet.Range(ROW_32_HEAD)
    p = -1
    For Each n In wksheet.Range(ROW_38_DATA)
        p = p + 1 ' index in list
        If c.Value = n.Value Then
            wksheet.Cells(c.Row, 44).Value = p
        End If
    Next n
Next c

'// create unit & header list
unitlist = "// Units, Col: 3" & vbCrLf & "unitlist:={"
headerlist = "// Header Labels, Col: 2" & vbCrLf & "headerlist:={"

'//
'// loop through each column and
'// create HPPL language compatible lists out of
'//
For j = 1 To MAX_PROPERTIES
    title = ""
    list = ""
    val = ""

    title = "// " & wksheet.Cells(2, j).Value & _
            " [" & wksheet.Cells(3, j).Value & "]" & ", Col:" & CStr(j) & _
            vbCrLf & wksheet.Cells(1, j).Value

    For i = OFFSET_START To OFFSET_START + MAX_ELEMENTS
        Select Case j
            Case 8
                '// use the indexed column value instead of plain text
                list = list & "," & wksheet.Cells(i, 40).Value
            Case 17
                '// use the indexed column value instead of plain text
                list = list & "," & wksheet.Cells(i, 41).Value
            Case 18
                '// use the indexed column value instead of plain text
                 list = list & "," & wksheet.Cells(i, 42).Value
            Case 27
                '// use the indexed column value instead of plain text
                list = list & "," & wksheet.Cells(i, 43).Value
            Case 32
                '// use the indexed column value instead of plain text
                list = list & "," & wksheet.Cells(i, 44).Value
            Case Else
                val = wksheet.Cells(i, j).Value
                Select Case j
                    Case 1, 2, 3, 6, 7, 8, 14, 15, 16, 17, 27, 30, 32
                        list = list & "," & Chr(34) & val & Chr(34)
                    Case Else
                        list = list & "," & val
                End Select
        End Select
    Next i

    '// strip first superflous comma
    list = Right(list, Len(list) - 1)
    ' Debug.Print title & ":={" & list & "};" & vbCrLf & vbCrLf
    list = title & ":={" & list & "};" & vbCrLf
    Print #filenum, list

    unitlist = unitlist & Chr(34) & wksheet.Cells(3, j).Value & Chr(34) & ","
    headerlist = headerlist & Chr(34) & wksheet.Cells(2, j).Value & Chr(34) & ","
Next j
'//
'// language dependent resource strings are starting at column(2) & row(130)
'//
langreslist = "// Language Labels, Col: 2" & vbCrLf & "langreslist:={"
For i = LANG_RES_START To LANG_RES_END
    langreslist = langreslist & Chr(34) & wksheet.Cells(i, 2).Value & Chr(34) & ","
Next i
'// strip last superflous comma
langreslist = Left(langreslist, Len(langreslist) - 1) & "};"
Print #filenum, langreslist & vbCrLf

'// strip first superflous comma
unitlist = Left(unitlist, Len(unitlist) - 1) & "};"
Print #filenum, unitlist & vbCrLf
'// strip first superflous comma
headerlist = Left(headerlist, Len(headerlist) - 1) & "};"
Print #filenum, headerlist & vbCrLf


'// replace repeated strings with equivalent number indexes from list
Print #filenum, distinctlist08 & vbCrLf
Print #filenum, distinctlist17 & vbCrLf
Print #filenum, distinctlist18 & vbCrLf
Print #filenum, distinctlist27 & vbCrLf
Print #filenum, distinctlist32 & vbCrLf

Print #filenum, "// Farben"
Print #filenum,
"distinct_colors:={#00EE00,#66AAFF,#FFAA00,#F3F300,#FFC1DC,#00DDBB,#FF93C2,#99BBAA,#B2AAFF,#B2F0FF,
#EFEFEF};" & vbCrLf

'// representation of periodic table and its atomic #no
Print #filenum, "// Element Position"
Print #filenum, "position:= ["
Print #filenum, "   [  1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  2 ],"
Print #filenum, "   [  3,  4,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  5,  6,  7,  8,  9, 10 ],"
Print #filenum, "   [ 11, 12,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 13, 14, 15, 16, 17, 18 ],"
Print #filenum, "   [ 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36 ],"
Print #filenum, "   [ 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54 ],"
Print #filenum, "   [ 55, 56, 57, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86 ],"
Print #filenum, "   [ 87, 88, 89,104,105,106,107,108,109,110,111,112,113,114,115,116,117,118 ],"
Print #filenum, "   [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0 ],"
Print #filenum, "   [  0,  0,  0, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71,  0 ],"
Print #filenum, "   [  0,  0,  0, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99,100,101,102,103,  0 ]"
Print #filenum, "   ];"
Print #filenum, "// ********************************************************************************"
Print #filenum, "// ********************************************************************************"
Print #filenum, "// ** You can safely replace entire content above with this export"
Print #filenum, "// ** it may change the language of the application"
Print #filenum, "// ********************************************************************************"
Print #filenum, "// ********************************************************************************"

Close #filenum
Set wksheet = Nothing
'//
'// open the HP Prime structure compatible file
'//
RetVal = Shell("NOTEPAD.EXE " & path & "\" & filename, vbNormalFocus)

End Sub
```

HP Prime Source Code (hpprgm format)

```
#pragma mode( separator(.,;) integer(h64) )
// Name [-], Col:1
symbolname:={"H","He","Li","Be","B","C","N","O","F","Ne","Na","Mg","Al","Si","P","S","Cl","Ar","K","Ca","Sc",
"Ti","V","Cr","Mn","Fe","Co","Ni","Cu","Zn","Ga","Ge","As","Se","Br","Kr","Rb","Sr","Y","Zr","Nb","Mo","Tc","
Ru","Rh","Pd","Ag","Cd","In","Sn","Sb","Te","I","Xe","Cs","Ba","La","Ce","Pr","Nd","Pm","Sm","Eu","Gd","Tb","
Dy","Ho","Er","Tm","Yb","Lu","Hf","Ta","W","Re","Os","Ir","Pt","Au","Hg","Tl","Pb","Bi","Po","At","Rn","Fr","
Ra","Ac","Th","Pa","U","Np","Pu","Am","Cm","Bk","Cf","Es","Fm","Md","No","Lr","Rf","Db","Sg","Bh","Hs","Mt","
Ds","Rg","Cn","Uut","Fl","Uup","Lv","Uus","Uuo"};

// Element [-], Col:2
elementname:={"Hydrogen","Helium","Lithium","Beryllium","Boron","Carbon","Nitrogen","Oxygen","Fluorine","Neon
","Sodium","Magnesium","Aluminium","Silicon","Phosphorus","Sulfur","Chlorine","Argon","Potassium","Calcium","
Scandium","Titanium","Vanadium","Chromium","Manganese","Iron","Cobalt","Nickel","Copper","Zinc","Gallium","Ge
rmanium","Arsenic","Selenium","Bromine","Krypton","Rubidium","Strontium","Yttrium","Zirconium","Niobium","Mol
ybdenum","Technetium","Ruthenium","Rhodium","Palladium","Silver","Cadmium","Indium","Tin","Antimony","Telluri
um","Iodine","Xenon","Caesium","Barium","Lanthanum","Cerium","Praseodymium","Neodymium","Promethium","Samariu
m","Europium","Gadolinium","Terbium","Dysprosium","Holmium","Erbium","Thulium","Ytterbium","Lutetium","Hafniu
m","Tantalum","Tungsten","Rhenium","Osmium","Iridium","Platinum","Gold","Mercury","Thallium","Lead","Bismuth"
,"Polonium","Astatine","Radon","Francium","Radium","Actinium","Thorium","Predactinium","Uranium","Neptunium",
"Plutonium","Americium","Curium","Berkelium","Californium","Einsteinium","Fermium","Mendelevium","Nobelium","
Lawrencium","Rutherfordium","Dubnium","Seaborgium","Bohrium","Hassium","Meitnerium","Darmstadtium","Roentgeni
um","Copernicium","Ununtrium","Flerovium","Ununpentium","Livermorium","Ununseptium","Ununoctium"};

// Atomic Weight [u], Col:3
atomicmass:={"1.008 (1.00784-1.00811)","4.002602","6.94 (6.938-6.997)","9.01218331","10.81 (10.806-
10.821)","12.011 (12.0096-2.0116)","14.0067 (14.00643-14.00728)","15.999 (15.99903-
15.99977)","18.998403163","20.1797","22.98976928","24.305 (24,304-24.307)","26.9815385","28.085 (28.084-
28.086)","30.973761998","32.06 (32.059-32.076)","35.45 (35.446-
35.457)","39.948","39.0983","40.078","44.955908","47.867","50.9415","51.9961","54.938044","55.845","58.933194
","58.6934","63.546","65.38","69.723","72.630","74.921595","78.971","79.904 (79.901-
79.907)","83.798","85.4678","87.62","88.90584","91.224","92.90637","95.95","98.9063","101.07","102.90550","10
6.42","107.8682","112.414","114.818","118.71","121.76","127.60","126.90447","131.293","132.90545196","137.327
","138.90547","140.116","140.90766","144.242","146.9151","150.36","151.964","157.25","158.92535","162.5","164
.93033","167.259","168.93422","173.054","174.9668","178.49","180.94788","183.84","186.207","190.23","192.217"
,"195.084","196.966569","200.592","204.38 (204.382-
204.385)","207.2","208.9804","209.98","209.9871","222","223.0197","226.0254","227.0278","232.0377","231.03588
","238.02891","237.0482","244.0642","243.061375","247.0703","247","15.1","8.84","257.0951","258","259","266",
"261.1087","262.1138","263.1182","262.1229","265, 269,
270","268","281","277","277","287","289","288","293","292","294"};

// Mol Weight [g.mol-1], Col:4
molmasse:={1,4,6.9,9,10.81,12.011,14,16,19,20.2,23,24.3,27,28.1,31,32.1,35.5,39.9,39.1,40.1,45.0,47.9,50.9,52
.0,54.9,55.8,58.9,58.7,63.5,65.4,69.7,72.6,74.9,79,79.9,83.8,85.5,87.6,88.9,91.2,92.9,95.9,98.9,101.1,102.9,1
06.4,107.9,112.4,114.8,118.7,121.8,127.6,126.9,131.3,132.9,137.3,138.9,140.1,140.9,144.2,145,150.4,152,157.3,
158.9,162.5,164.9,167.3,168.9,173,175,178.5,180.9,183.9,186.2,190.2,192.2,195.1,197,200.6,204.4,207.2,209,209
,210,222,223,226,227,232,231,238,237,244,243,247,247,251,252,257,258,259,260,261,262,263,262,265,266,269,280,
285,287,289,288,293,292,294};

// Density [g/cm^3], Col:5
density:={0.0899,0.1785,0.534,1.848,2.460,2.26,0.1250,1.429,1.6965,0.900,0.968,1.738,2.7,2.336,1.823,2.07,3.2
14,1.784,0.856,1.55,2.985,4.507,6.11,7.14,7.47,7.874,8.9,8.908,8.92,7.14,5.904,5.323,5.727,4.819,3.12,3.7491,
1.532,2.63,4.472,6.511,8.57,10.28,11.5,12.37,12.45,12.023,10.49,8.65,7.31,7.31,6.697,6.24,4.94,5.8982,1.879,3
.51,6.146,6.773,6.64,7.01,7.264,7.353,5.244,7.901,8.219,8.551,8.795,9.066,9.321,6.57,9.841,13.31,16.65,19.25,
21.02,22.59,22.56,21.09,19.3,13.534,11.85,11.34,9.78,9.196,0,9.73,0,5.5,10.07,11.724,15.37,19.05,20.45,19.816
,13.67,13.51,14.78,15.1,8.84,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// Oxidation States [-], Col:6
oxidstates:={"+1, 0, -1","0","+1","2","3","2, 4","-3, -2, -1, 1, 2, 3, 4, 5","-2, -1, 0, +1, +2","-1","-
2","1","2","1, 2, 3","-4, (2) +4","±3, 4, 5","±2, 4, 6","±1, 3, 4, 5, 6, 7","-","1","(+1), +2","3","+2, +3,
+4","+5, +4 ,+3 ,+2","6, 3, 2","1, 2, 3, 4, (5), 6, 7","2, 3, 4, 5, 6","2, 3","2, seltener -1, 0, 1, 3,
4","1, 2","2","3","-4, 2, 4","-3, 3, 5","±2, 4, 6","±1, 3, 5, 7","-","+1","+2","+3","4, 2","2, 5","2, 3, 4,
5, 6","-3 bis 7","2, 3, 4, 6, 8","0, +1, +2, +3, +4","0, +2, +4","+1, +2, +3","2","3, 1","(-4), 4, 2","-3, 3,
5","-2, (±1), 2, 4, 6","±1, 3, 5, 7","-","+1","+2","3","3, 4","3, 4, 2","3, 4","3","2, 3","2, 3","2, 3","4,
3","3","3","3","2, 3, 4","2, 3","3","4","5","6, 5, 4, 3, 2","-3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7","-2, 0, 2,
3, 4, 6, 8","-1, 0, 1, 2, 3, 4, 6","0, +2, +4, +6","-1, 0, +1, +2, +3, +5","1, 2, 4","1, 3","2, 4","(-3) 1,
3, 5","(-2), 2, 4, 6","±1, 3, 5, 7","-","+1","2","3","4, 3, 2","5","+3, +4, +5, +6","+3, +4, +5, +6, +7","+3,
+4, +5, +6, (+7)","+2, +3, +4, +5, +6, (+7)","(+2), +3, +4","+3, +4","(+2), +3, (+4)","+2, +3, (+4)","+2,
+3","+2, +3","+2, +3","+3","-","-","-","-","-","-","-","-","-","-","-","-","-","-","-"};

// Elektron Configuration [-], Col:7
elektronconfig:={"1s1","1s2","[He] 2s1","[He] 2s2","[He] 2s2 2p1","[He] 2s2 2p2","[He] 2s2 2p3","[He] 2s2
2p4","[He] 2s2 2p5","[He] 2s2 2p6","[Ne] 3s1","[Ne] 3s2","[Ne] 3s2 3p1","[Ne] 3s2 3p2","[Ne] 3s2 3p3","[Ne]
3s2 3p4","[Ne] 3s2 3p5","[Ne] 3s2 3p6","[Ar] 4s1","[Ar] 4s2","[Ar] 3d1 4s2","[Ar] 3d2 4s2","[Ar] 3d3
4s2","[Ar] 3d5 4s1","[Ar] 3d5 4s2","[Ar] 3d6 4s2","[Ar] 3d7 4s2","[Ar] 3d8 4s2, [Ar] 3d9 4s1","[Ar] 3d10
4s1","[Ar] 3d10 4s2","[Ar] 3d10 4s2 4p1","[Ar] 3d10 4s2 4p2","[Ar] 3d10 4s2 4p3","[Ar] 3d10 4s2 4p4","[Ar]
3d10 4s2 4p5","[Ar] 3d10 4s2 4p6","[Kr] 5s1","[Kr] 5s2","[Kr] 4d1 5s2","[Kr] 4d2 5s2","[Kr] 4d4 5s1","[Kr]
4d5 5s1","[Kr] 4d5 5s2","[Kr] 4d7 5s1","[Kr] 4d8 5s1","[Kr] 4d10 5s0","[Kr] 4d105s1","[Kr] 4d10 5s2","[Kr]
4d10 5s2 5p1","[Kr] 4d10 5s2 5p2","[Kr] 4d10 5s2 5p3","[Kr] 4d10 5s2 5p4","[Kr] 4d10 5s2 5p5","[Kr] 4d10 5s2
5p6","[Xe] 6s1","[Xe] 6s2","[Xe] 5d1 6s2","[Xe] 4f1 5d1 6s2","[Xe] 4f3 6s2","[Xe] 4f4 6s2","[Xe] 4f5
6s2","[Xe] 4f6 6s2","[Xe] 4f7 6s2","[Xe] 4f7 5d1 6s2","[Xe] 4f9 6s2","[Xe] 4f10 6s2","[Xe] 4f11 6s2","[Xe]
4f12 6s2","[Xe] 4f13 6s2","[Xe] 4f14 6s2","[Xe] 4f14 5d1 6s2","[Xe] 4f14 5d2 6s2","[Xe] 4f14 5d3 6s2","[Xe]
4f14 5d4 6s2","[Xe] 4f14 5d5 6s2","[Xe] 4f14 5d6 6s2","[Xe] 4f14 5d7 6s2","[Xe] 4f14 5d9 6s1","[Xe] 4f14 5d10
6s1","[Xe] 4f14 5d10 6s2","[Xe] 4f14 5d10 6s2 6p1","[Xe] 4f14 5d10 6s2 6p2","[Xe] 4f14 5d10 6s2 6p3","[Xe]
4f14 5d10 6s2 6p4","[Xe] 4f14 5d10 6s2 6p5","[Xe] 4f14 5d10 6s2 6p6","[Rn] 7s1","[Rn] 7s2","[Rn] 6d1
7s2","[Rn] 6d2 7s2","[Rn] 5f2 6d1 7s2","[Rn] 5f3 6d1 7s2","[Rn] 5f4 6d1 7s2","[Rn] 5f6 7s2","[Rn] 5f7
7s2","[Rn] 5f7 6d1 7s2","[Rn] 5f9 s2","[Rn] 5f10 7s2","[Rn] 5f11 7s2","[Rn] 5f12 7s2","[Rn] 5f13 7s2","[Rn]
5f14 7s2","[Rn] 5f14 6d1 7s2, [Rn] 5f14 7s2 7p1","[Rn] 5f14 6d2 7s2","[Rn] 5f14 6d3 7s2","[Rn] 5f14 6d4
7s2","[Rn] 5f14 6d5 7s2","[Rn] 5f14 6d6 7s2","[Rn] 5f14 6d7 7s2","[Rn] 5f14 6d8 7s2","[Rn] 5f14 6d10
7s1","[Rn] 5f14 6d10 7s2","[Rn] 5f14 6d10 7s2 7p1","[Rn] 5f14 6d10 7s2 7p2","[Rn] 5f14 6d10 7s2 7p3","[Rn]
5f14 6d10 7s2 7p4","[Rn] 5f14 6d10 7s2 7p5","[Rn] 5f14 6d10 7s2 7p6"};

// Phase [-], Col:8
aggregatstate:={1,1,2,2,2,2,1,1,1,1,2,2,2,2,2,2,1,1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1,2,2,2,2,2,2,2,2,2,2
,2,2,2,2,2,2,1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,2,2,2,2,2,1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
2,2,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4};

// Melting Point [°C], Col:9
meltingpoint:={-259.14,-272.2,181,1278,2076,3370,-210.1,-218.3,-219.62,-
248.59,97.72,650,660.2,1410,44.2,115.21,-101.5,-
189.3,64,842,1541,1668,1910,1907,1246,1538,1495,1455,1084.62,419.53,29.76,938.3,0,221,-7.3,-
157.36,39.31,777,1526,1857,2477,2623,2157,2334,1964,1554.9,961.78,321.07,156.5985,231.93,630.63,449.51,113.7,
-
111.7,28.44,727,920,795,935,1024,1080,1072,826,1312,1356,1407,1461,1529,1545,824,1652,2233,3017,3422,3186,313
0,2466,1768.3,1064.18,-38.83,304,327.43,271.3,254,302,-
71,27,700,1050,1755,1568,1133,639,639.4,1176,1340,986,900,860,852,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// Boiling Point [°C], Col:10
boilingpoint:={-252,-269,1347,2970,3930,4827,-196,-183,-188,-246,890,1110,2470,3260,280,445,-34.6,-
186,774,1487,2730,3260,3407,2482,2100,3000,2900,2730,2595,907,2400,2830,613,685,58.5,-
152,688,1380,2930,4377,4744,4612,4265,4150,3727,2960,2210,765,2000,2620,1635,990,184,-
108,690,1637,3470,3470,3130,3030,3000,1900,1440,3000,3123,2600,2600,2900,1950,1430,3330,4603,5420,5930,5630,5
000,4130,3827,2970,357,1460,1744,1560,962,337,-
61.8,677,1737,3300,4788,0,3930,3902,3230,2607,3110,0,0,996,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// Heat of Vaporization [kJ/mol], Col:11
boilingenthalpy:={0.9,0.084,136,309,508,0,5.58,5.58,6.32,1.9,97.4,132,284,383,51.9,45,20.4,6.52,79.1,153,310,
457,444,347,225,354,390,379,305,115,256,330,32.4,95.5,30,9.03,69,141,390,591,694,617,550,619,531,380,254,100,
225,290,193,114,41.6,12.6,66.1,149,400,398,331,289,290,192,176,301,391,280,251,280,247,159,414,648,753,774,70
7,678,564,510,342,58.2,162,177,179,100,40,16.4,65,125,400,530,470,417.1,1420,325,238,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0};

// Heat of Fusion [kJ/mol], Col:12
meltingenthalpy:={0.558,0.02,3,7.95,50,0,0.36,0.222,0.2552,0.34,2.6,8.7,10.7,50.66,0.64,1.713,3.2,1.18,2.334,
8.54,16,18.7,21.5,16.93,13.2,13.8,17.2,17.7,13.3,7.4,5.59,31.8,27.7,5.4,5.8,1.64,2.19,8,11.4,16.9,26.8,36,23,
25.7,21.7,16.7,11.3,6.2,3.26,7.03,19.7,17.5,7.76,2.3,2.09,8.0,6.2,5.5,6.9,7.1,7.7,8.6,9.2,10.0,10.8,11.06,17,
19.9,16.8,7.6,22,25.5,36,35.2,33,31.8,26,19.6,12.55,2.37,4.2,4.85,10.9,13,6,2.89,2,8,14,16,15,15.5,39.91,11.4
8,14.4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// Heat Capacity [J/(kg.K)], Col:13
heatcapacity:={14304,5193,3482,1825,1260,709,
427,1040,920,0,0,1230,1023,897,703,685.6,736,480,0,757.8,647.3,586,523,489,449,479.5,449,421,444,385,388,371,
308.3,328.5,0,0,0,0,0,0,270,0,0,0,0,243,0,235,0,233,0,0,0,0,0,0,0,0,0,0,190,0,0,0,0,0,0,0,0,0,0,140,140,138
,137,130,0,130,128,140,129,131,0,0,0,0,0,0,0,0,116,0,130,0.11,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
};

// Group [-], Col:14
group:={"1","18","1","2","13","14","15","16","17","18","1","2","13","14","15","16","17","18","1","2","3","4",
"5","6","7","8","9","10","11","12","13","14","15","16","17","18","1","2","3","4","5","6","7","8","9","10","11
","12","13","14","15","16","17","18","1","2","3","La","La","La","La","La","La","La","La","La","La","La","La",
"La","La","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","1","2","3","Ac","Ac","Ac","Ac
","Ac","Ac","Ac","Ac","Ac","Ac","Ac","Ac","Ac","Ac","4","5","6","7","8","9","10","11","12","13","14","15","16
","17","18"};

// Period [-], Col:15
period:={"1","1","2","2","2","2","2","2","2","2","3","3","3","3","3","3","3","3","4","4","4","4","4","4","4",
"4","4","4","4","4","4","4","4","4","4","4","5","5","5","5","5","5","5","5","5","5","5","5","5","5","5","5","
5","5","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6","6
","6","6","6","6","6","6","7","7","7","7","7","7","7","7","7","7","7","7","7","7","7","7","7","7","7","7","7"
,"7","7","7","7","7","7","7","7","7","7","7"};

// Block [-], Col:16
block:={"s","s","s","s","p","p","p","p","p","p","s","s","p","p","p","p","p","p","s","s","d","d","d","d","d","
d","d","d","d","d","p","p","p","p","p","p","s","s","d","d","d","d","d","d","d","d","d","d","p","p","p","p","p
","p","s","s","d","f","f","f","f","f","f","f","f","f","f","f","f","f","f","d","d","d","d","d","d","d","d","d"
,"p","p","p","p","p","p","s","s","d","f","f","f","f","f","f","f","f","f","f","f","f","f","f","d","d","d","d",
"d","d","d","d","d","d","d","d","d","d","d"};

// Category [-], Col:17
family:={5,1,2,3,4,5,5,5,6,1,2,3,7,4,5,5,6,1,2,3,8,8,8,8,8,8,8,8,8,8,7,4,4,4,6,1,2,3,8,8,8,8,8,8,8,8,8,8,7,7,
4,4,6,1,2,3,8,9,9,9,9,9,9,9,9,9,9,9,9,9,9,8,8,8,8,8,8,8,8,8,7,7,7,7,6,1,2,3,8,10,10,10,10,10,10,10,10,10,10,1
0,10,10,10,8,8,8,8,8,11,11,11,8,11,11,11,11,11,11};

// Crystal Structure [-], Col:18
crystal:={1,1,2,3,4,5,6,7,1,8,2,6,8,9,10,10,10,8,2,8,6,6,2,2,11,11,6,8,8,6,11,9,12,11,10,8,2,8,6,6,2,2,6,6,8,
8,8,6,13,13,14,14,10,8,2,2,6,8,6,6,11,14,2,6,6,6,6,6,6,8,6,6,2,2,6,6,8,8,8,15,6,8,14,11,11,8,11,2,8,8,13,10,1
0,16,6,6,6,6,7,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11};
```

```
// Atomic Volume [cm^3/mol], Col:19
atomicvolume:={11.42,21.00,13.02,4.85,4.39,5.31,13.54,17.336,11.20,13.23,23.78,14.00,10.00,12.06,17.02,15.53,
17.39,22.56,45.94,26.20,15.00,10.64,8.32,7.23,7.35,7.09,6.67,6.59,7.11,9.16,11.8,13.63,12.95,16.42,19.78,27.9
9,55.76,33.94,19.88,14.02,10.83,9.38,8.53,8.17,8.28,8.56,10.27,13.00,15.76,16.29,18.19,20.46,25.72,35.92,70.9
4,38.16,22.39,20.69,20.80,20.59,20.10,19.98,28.97,19.90,19.30,19.01,18.74,18.46,19.1,24.84,17.78,13.44,10.85,
9.47,8.86,8.42,8.52,9.09,10.21,14.09,17.22,18.26,21.31,22.97,0,50.50,0,41.09,22.55,19.80,15.18,12.49,11.59,12
.29,1.778,18.05,16.84,16.50,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// Atomic Radius [pm], Col:20
atomradius:={53,31,167,112,84,67,56,48,42,38,190,145,118,111,98,88,79,71,243,194,184,176,171,166,161,156,152,
149,145,142,136,125,114,103,94,88,265,219,212,206,164,190,185,178,173,169,165,161,156,145,133,123,115,108,298
,253,195,185,247,206,205,238,231,233,225,228,226,226,222,222,217,208,200,193,188,185,180,177,174,171,156,154,
143,135,127,120,0,215,195,180,180,138.5,130,151,184,174.3,170,186,203,198,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
};

// Covalent Radius [pm], Col:21
covalentradii:={31,28,128,96,82,76,71,66,71,58,154,141,121,111,107,102.5,102,106,203,176,170,160,153,139,139,
123,126,124,132,122,122,122,119,120,120,116,220,195,190,206,137,154,147,146,142,139,145,144,144,139,139,138,1
39,140,244,215,207,204,203,201,199,198,198,196,194,192,192,189,190,187,187,150,138,162,159,128,141,136,136,13
2,145,146,148,140,150,150,260,221,215,206,200,142,0,187,0,0,0,225,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// Van-der-Waals-Radius [pm], Col:22
vanderwaals:={120,140,182,153,192,170,155,152,147,154,227,173,184,210,180,180,175,188,275,231,0,0,0,0,0,0,0,1
63,140,139,187,211,185,190,185,202,303,249,0,0,0,0,0,0,0,163,172,158,193,217,206,206,198,216,343,268,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,175,166,155,196,202,0,197,202,220,348,283,0,0,0,0,0,0,228.5,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// Thermal Conductivity [W/(m*K)], Col:23
thermalconductivity:={0.1805,0.1513,85,25,27,0,0.02583,0.02658,0.0279,0.0491,140,160,235,150,0.236,0.205,0.00
89,0.01772,100,200,16,22,0,449,7.8,80,100,91,400,120,29,60,50,0.52,0.12,0.00949,58,35,17,22.7,54,139,51,120,1
50,72,430,97,81.6,67,24,3,0.449,0.00569,36,18,13,11,13,17,15,13,14,11,11,11,16,15,16.8,39,16,23,57,170,48,88,
150,72,320,8.3,46,35,8,20,2,0.00364,0,19,12,54,47,27.6,6.40,6.74,10,10,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0};

// Electrical Conductivity [10^6 A/(V*m)], Col:24
electroconductivity:={0,0,10.6,190,0.0001,0,0,0,0,0,21,22.7,37.7,1000,0,0,0,0,14.3,29.4,1.81,2.5,5,7.87,0.694
,10,16.7,13.9,58.1,16.7,7.44,0.0000021,3.03,0,0,0,7.52,7.41,1.66,2.36,6.58,18.2,4.54,14.1,23.3,9.26,61.35,14.
3,12.5,8.69,2.5,10000,0,0,4.76,2.94,1.626,1.35,1.43,1.56,1.33,1.06,1.11,0.763,0.870,1.08,1.23,1.16,1.477,4.0,
1.72,3.12,7.61,18.52,5.56,10.9,19.7,9.43,45.5,1.04,6.67,4.76,0.769,2.5,0,0,0,1,0,6.67,5.56,2.24,0.00000082,0.
68,0.000147,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// First Ionisation Energy [kJ/mol], Col:25
firstionenergy:={1312,2372,520.2,1825,800,1086,1402,1313.9,1681,2080,495.8,737.7,577.5,786.5,1011.8,999.6,125
1.2,1520.8,418.8,589.8,633.1,658.8,650.9,652.9,717.3,762.5,760.4,737.1,745.5,906.4,578.8,762,947,941,1139.9,1
350.8,403,549.5,600,640.1,652.1,684.3,702,710.2,719.7,804.4,731.0,867.8,558.3,708.6,834,869.3,1008.4,1170.4,3
75.7,502.9,538.1,534.4,527,533.1,540,544.5,547.1,593.4,565.8,573.0,581,589,596.7,603.4,523.5,658.5,761,770,76
0,840,880,870,890.1,1007.1,589.4,715.6,703,812.1,920,1037,380,509.3,499,578,568,597.6,604.5,584.7,578,581,0,0
,613.84,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// Elektronegativity [-], Col:26
elektronegativity:={2.2,5.2,0.98,1.57,2.04,2.55,3.04,3.44,4.0,0,0.93,1.31,1.61,1.90,2.19,2.58,3.16,0,0.82,1,1
.36,1.54,1.63,1.66,1.55,1.83,1.88,1.91,1.9,1.65,1.81,2.01,2.18,2.55,2.96,0,0.82,0.95,1.22,1.33,1.6,2.16,2.1,2
.2,2.28,2.2,1.93,1.69,1.78,1.96,2.05,2.1,2.66,0,0.79,0.89,1.1,1.12,1.13,1.14,0,1.17,0,1.2,0,1.22,1.23,1.24,1.
25,0,1,1.3,1.5,1.7,1.9,2.2,2.2,2.2,2.4,1.9,1.8,1.8,1.9,2,2.2,0,0.7,0.9,1.1,1.3,1.5,1.7,1.3,1.3,1.3,1.3,1.3,1.
3,1.3,1.3,1.3,1.3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

// Oxide [-], Col:27
oxide:={3,1,2,3,4,4,5,6,6,6,2,7,3,3,4,5,6,6,2,2,8,3,6,6,5,3,3,9,9,6,6,3,3,5,6,6,2,2,8,6,6,6,5,6,3,9,3,9,6,6,6
,4,6,6,2,7,8,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,9,3,9,6,3,6,3,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,
6,6,6,6,6,6,6,6,6,6,6,6,6};

// Discovery [Year], Col:28
discovery:={1766,1895,1817,1798,1808,0,1772,1774,1886,1898,1807,1755,1827,1824,1669,0,1774,1894,1807,1808,187
8,1791,1801,1797,1774,0,1735,1751,0,0,1875,1886,0,1817,1826,1898,1861,1808,1794,1789,1801,1782,1937,1844,1803
,1803,0,1817,1863,0,0,1782,1811,1898,1860,1808,1839,1803,1885,1885,1945,1879,1901,1880,1843,1886,1878,1842,18
79,1878,1907,1923,1802,1783,1925,1803,1803,1735,0,0,1861,0,0,1898,1940,1900,1939,1898,1899,1828,1913,1789,194
0,1940,1944,1944,1949,1950,1952,1952,1955,1958,1961,1964,1967,1974,1981,1984,1982,1994,1994,1996,1994,2009,20
12,2011,2006,2011};

// Color Index (META INFO) [-], Col:29
colorindex:={1,2,3,4,7,1,1,1,8,2,3,4,6,7,1,1,8,2,3,4,5,5,5,5,5,5,5,5,5,5,6,7,7,1,8,2,3,4,5,5,5,5,5,5,5,5,5,5,
6,6,7,7,8,2,3,4,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,5,5,5,5,5,5,5,5,5,6,6,6,7,8,2,3,4,10,10,10,10,10,10,10,10,10,10
,10,10,10,10,10,5,5,5,5,5,11,11,11,5,11,11,11,11,11,11};

// CAS Number [-], Col:30
CASnumber:={"1333-74-0","7440-59-7","7439-93-2","7440-41-7","7440-42-8","7440-44-0","7727-37-9","7782-44-
7","7782-41-4","7440-01-9","7440-23-5","7439-95-4","7429-90-5","7440-21-3","7723-14-0","7704-34-9","7782-50-
5","7440-37-1","7440-09-7","7440-70-2","7440-20-2","7440-32-6","7440-62-2","7440-47-3","7439-96-5","7439-89-
6","7440-48-4","7440-02-0","7440-50-8","7440-66-6","7440-55-3","7440-56-4","7440-38-2","7782-49-2","7726-95-
6","7439-90-9","7440-17-7","7440-24-6","7440-65-5","7440-67-7","7440-03-1","7439-98-7","7440-26-8","7440-18-
8","7440-16-6","7440-05-3","7440-22-4","7440-43-9","7440-74-6","7440-31-5","7440-36-0","13494-80-9","7553-56-
2","7440-63-3","7440-46-2","7440-39-3","7439-91-0","7440-45-1","7440-10-0","7440-00-8","7440-12-2","7440-19-
9","7440-53-1","7440-54-2","7440-27-9","7429-91-6","7440-60-0","7440-52-0","7440-30-4","7440-64-4","7439-94-
3","7440-58-6","7440-25-7","7440-33-7","7440-15-5","7440-04-2","7439-88-5","7440-06-4","7440-57-5","7439-97-
6","7440-28-0","7439-92-1","7440-69-9","7440-08-6","7440-68-8","10043-92-2","7440-73-5","7440-14-4","7440-34-
8","7440-29-1","7440-13-3","7440-61-1","7439-99-8","7440-07-5","7440-35-9","7440-51-9","7440-40-6","7440-71-
3","7429-92-7","7440-72-4","7440-11-1","10028-14-5","22537-19-5","53850-36-5","53850-35-4","54038-81-
2","54037-14-8","54037-57-9","54038-01-6","54083-77-1","54386-24-2","54084-26-3","54084-70-7","54085-16-
4","54085-64-2","54100-71-9","54101-14-3","54144-19-3"};

// Mass on Eearth Shell [ppm], Col:31
earthmass:={1500,0.004,60,5.3,16,870,300,494000,280,0.005,26400,19400,75700,258000,900,480,1900,3.6,24100,339
00,5.1,4100,410,190,850,47000,37,150,100,120,14,5.6,5.5,0.8,6,0.000019,29,0.014,26,0.021,19,14,0.000000000000
0012,0.02,0.001,0.011,0.079,0.3,0.1,35,0.65,0.01,0.06,0.000009,6.5,260,17,43,5.2,22,0.0000000000000015,6,0.09
9,5.9,0.85,4.3,1.1,2.3,0.19,2.5,0.7,4.2,8,64,0.001,0.01,0.001,0.005,0.004,0.4,0.29,18,0.2,0.000000000021,0.00
0000000000000000003,0.000000000061,0.0000000000000000013,0.000000000095,0.00000000000006,11,0.00000009,3.2,0.
00000000000004,0.0000000000000002,0,0,0,0,0,0,0,0,0,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14};

// Look Alike [-], Col:32
lookalike:={0,1,2,3,4,5,1,1,6,1,7,7,8,9,10,11,12,1,7,7,7,13,14,13,15,16,17,18,19,20,7,21,22,23,24,25,7,26,7,7
,27,28,29,26,26,26,30,29,31,31,31,32,33,25,34,3,7,7,35,35,36,7,7,7,7,7,7,7,37,7,7,38,39,40,21,41,7,42,43,7,7,
44,45,8,36,25,46,47,8,7,48,7,8,49,50,50,51,52,46,46,46,46,46,46,46,46,46,46,46,46,46,46,46,46,46,46,46};

// Language Labels, Col: 2
langreslist:={"Help","Periodic Table","Atom number","Click=Select element ; ENTER=Element properties ;
HOME=Table ; Drag=Move table ; Mol menu=Molar mass","Z= Menu=Search Atom N° ; Plot=Statistical Plot","By
Mickaël Nicotera","Rev. by Miguel Toro","Synthetic element","General","Known since ancient
world","Atomic","Physical","Chemical","Arrow up","Arrow down","for more…","Choose the property to plot:","1.)
Earth mass abundance","2.) Atomic volume","3.) Density","4.) Atomic radius","5.) Melting point","6.) Boiling
point","Earth","Volume","Density","Radius","Melt","Boil","List","Plot","","Help","About","Quit","Property to
Plot","Connect dots"};

// Units, Col: 3
unitlist:={"-","-","u","g.mol-1","g/cm^3","-","-","-","°C","°C","kJ/mol","kJ/mol","J/(kg.K)","-","-","-","-
","-","cm^3/mol","pm","pm","pm","W/(m*K)","10^6 A/(V*m)","kJ/mol","-","-","Year","-","-","ppm","-"};

// Header Labels, Col: 2
headerlist:={"Name","Element","Atomic Weight","Mol Weight","Density","Oxidation States","Elektron
Configuration","Phase","Melting Point","Boiling Point","Heat of Vaporization","Heat of Fusion","Heat
Capacity","Group","Period","Block","Category","Crystal Structure","Atomic Volume","Atomic Radius","Covalent
Radius","Van-der-Waals-Radius","Thermal Conductivity","Electrical Conductivity","First Ionisation
Energy","Elektronegativity","Oxide","Discovery","Color Index (META INFO)","CAS Number","Mass on Eearth
Shell","Look Alike"};

distinct_aggregatstate:={"gas","solid","liquid","-"};

distinct_family:={"Noble gases","Alkali metals","Alkaline earth metals","Post-
transition","Nonmetals","Halogens","Metalloids","Transition metals","Lanthanide","Actinide","unknown"};

distinct_crystal:={"diamagnetisch","body-centered cubic","hexagonal (dichtest
gepackt)","rhomboedric","hexagonal, face-centered cubic","hexagonal","cubic","face-centered
cubic","diamantstruktur","orthorhombisch","-","trigonal (gray)","tetragonal","trigonal","liquid","monoklin"};

distinct_oxide:={"none","strong basic","amphoter","light acid","strong acid","-","basic","mild basic","light
basic"};

distinct_lookalike:={"colorless gas","silvery white/gray","white-gray metallic","black","black (G), colorless
(D), yelllow-brown (L)","very pale yellow gas","silvery white","silvery","darkgray, bluish color","white-
beige (W), darkred (R), black (S)","yellow","yellow-green","silvery metallic","metallicgray bluish
shimmery","silvery metallic (iron white)","metallic shimmery with a gray color","metallic with a bluish-gray
color","shimmery, metallic, silvery","salmonpink metallic","bluish pale gray","grayish white","metallic gray,
yellow or black","gray, shimmery","gas: red-brown, liquid: red-brown, solid: metallic
shimmery","colorless","silvery white metallic","gray metallic shimmery","gray metallic","silvery gray
metallic","whiteshimmery, metallic","silvery shimmery gray","silverwhite, metallic shimmery","gas: dark-
violet, solid: gray, shimmery","silverwhite shimmery, with low acid golden yellow","silvery white, yellowish
color","metallic","silvery gray","metallicgray","gray","grayish white, shimmery","bluish gray","gray-
white","metallic yellow","bluish white","shimmery redish white","unknown","silvery-white-metallic","bright,
silvery, metallic shimmery","silvery metal","silvery-white metal","silverwhite","silveryes metal"};

// Farben
distinct_colors:={#00EE00,#66AAFF,#FFAA00,#F3F300,#FFC1DC,#00DDBB,#FF93C2,#99BBAA,#B2AAFF,#B2F0FF, #EFEFEF};

// Element Position
position:= [
[ 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2 ],
[ 3, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 6, 7, 8, 9, 10 ],
[ 11, 12, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 13, 14, 15, 16, 17, 18 ],
[ 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36 ],
[ 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54 ],
[ 55, 56, 57, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86 ],
[ 87, 88, 89,104,105,106,107,108,109,110,111,112,113,114,115,116,117,118 ],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ],
[ 0, 0, 0, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 0 ],
[ 0, 0, 0, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99,100,101,102,103, 0 ]
];
// ****************************************************************************
// ****************************************************************************
// ** You can safely replace entire content above with this export
// ** it may change the language of the application
// ********************************************************************************
// ********************************************************************************


// see Erich Rechlins pages...
// see http://www.hpcalc.org/details.php?id=7556 for HP Elements
// see http://www.hpcalc.org/details.php?id=7530 for WaitLlab
Credit:= {"Only to be on par with the HP48SX (as extension card)",
"that was later integrated on the HP48GX",
"while using the power and screen tech of the Prime",
"based on & inspirated by:",
" ҉ Elements ҉ HP Prime by Mickaël Nicotera",
"Revised by Miguel Toro",
"Message Loop adapted from WaitLab jfs 2014-06-04",
"HP for there outstanding calculator hardware",
"...and Joseph K. Horn","Data from de.wikipedia.org Dec. 2014"};
//
// function declarations
//
DoLoop(Gr);
Event_InitApp(Gr);
Event_Redraw(Gr);
Event_KeyPress(Gr,KeyCode);
Event_MenuClick(Gr,Index);
Event_MouseDown(Gr,x,y);
Event_MouseUp(Gr,x,y,typ);
Event_MouseClick(Gr,x,y);
Event_MouseLongClick(Gr,x,y);
Event_DragStart(Gr,x,y);
Event_DragMove(Gr,x,y);
Event_DragEnd(Gr,x,y);
//
build_table(Gr);
select_element(Gr,posx,posy);
show_element(Gr,sel,y);
show_list(Gr,sort_alpha,posx,posy);
show_about(Gr);
show_plot(Gr,prop,sel,con_dots);
prop_plot();
selected_plot(x,y);
//
// Key Codes Constants
//
K_APPS=0, K_SYMB=1, K_ARROWUP=2, K_HELP=3, K_ESC=4, K_HOME=5;
K_PLOT=6, K_ARROWLEFT=7, K_ARROWRIGHT=8, K_VIEW=9, K_CAS=10;
K_NUM=11, K_ARROWDOWN=12, K_MENU=13, K_A=14, K_B=15, K_C=16;
K_D=17, K_E=18, K_DEL=19, K_F=20, K_G=21, K_H=22, K_I=23, K_J=25;
K_K=26, K_M=27, K_N=28, K_O=29, K_ENTER=30, K_P=31, K_Q=32, K_R=33;
K_S=34, K_T=35, K_ALPHA=36, K_U=37, K_V=38, K_W=39, K_X=40;
K_SHIFT=41, K_Y=42, K_Z=43, K_HASH=44, K_DOUBLEDOT=45, K_OFF=46;
K_NOTES=47, K_EQUAL=48, K_UNDERSCORE=49, K_ANS=50;
//
// Other Constants
//
HSIZE:=400; // Main Screen, Total image horizontal size
VSIZE:=560; // Property Pages, total vertical size
LSIZE:=1640; // Entire List, total vertical size
el_width:=22; // width of an element in main view
el_height:=21; // height of an element in main view
text_offset:=6;
MAX_ELEMENTS:=118;
LANG_ID:=0;
LIGHTBLUE:=rgb(0,124,240);
LIGHTGRAY:=rgb(213,213,213);
BLACK:=#000000;
WHITE:=#FFFFFF;
linH:=13;
main_window:=G1; // periodic table
property_window:=G2; // individual element pages
list_window:=G3; // full sortable list
about_window:=G4; // dedicated
plot_window:=G5; // plot (not sufficient accessible plotting App, not enough flexibility, and too much
dependency)
//
// Globals (to the app)
// true globals would have an export in front of
//
// Variables to manage the scrolling
MainWndBeg:=0;
MainWndEnd:=320;
MainOffset;
MainSmooth:=1;
```

```
MainDOffset:=0;

// Property Window
PropWndBeg:=0;
PropWndEnd:=320;
PropOffset;
PropSmooth:=1;
PropDOffset:=0;

// List Window
ListWndBeg:=0;
ListWndEnd:=320;
ListOffset;
ListSmooth:=1;
ListDOffset:=0;
LCurrSortOrder:=0;
new_order; // list

previous:=0; // last selected element
selected:=Z; // current selected element
selected_prop:=0;
connect_dots:=1;
appmenu:={langreslist(LANG_ID+30),langreslist(LANG_ID+31),langreslist(LANG_ID+32),langreslist(LANG_ID+33),lan
greslist(LANG_ID+34),langreslist(LANG_ID+35)};
//
// helpers
//
move_up_to();
move_down_to();
up_arrow(Gr,x,y,color);
left_arrow(Gr,x,y,color);
right_arrow(Gr,x,y,color);
down_arrow(Gr,x,y,color);
random_color();
center_text(txt,Gr,x_width,y);
GetTextExtent(txt,fsize);
ralign_text(txt,Gr,x,y);
chaos();
cross_point(Gr,x,y,color);
check_mark(Gr,x,y,color,on_off);
//
// main loop
//
export PTable()
begin
Event_InitApp(main_window);
DoLoop(main_window);
end;
//
// Start Subroutines
//
Event_InitApp(Gr)
begin
// create menu
drawmenu(appmenu);
// create initial screen
build_table(Gr);
// draw the screen first time
blit_p(G0,Gr,0,0,320,220);
end;
//
// refresh the screen (after each loop)
//
Event_Redraw(Gr)
begin
local i,j;
case
if (Gr==main_window) then
// draw main window
blit_p(G0,Gr,MainWndBeg,0,MainWndEnd,220);
drawmenu(appmenu);
end;
if (Gr==property_window) then
// draw the properties
blit_p(G0,Gr,0,PropWndBeg,320,PropWndEnd);
end;
if (Gr==list_window) then
// draw the list
blit_p(G0,Gr,0,ListWndBeg,320,ListWndEnd);
end;
if (Gr==about_window) then
// draw the credits
blit_p(G0,Gr,0,0,320,240);
end;
```

```
if (Gr==plot_window) then
// draw the plot
blit_p(G0,Gr,0,0,320,240);
end;
end;
end;
//
//
//
Event_MouseDown(Gr,x,y)
begin
// stub to satisfy compiler
end;
//
//
//
Event_MouseUp(Gr,x,y)
begin
// stub to satisfy compiler
end;
//
// put here code you want to execute
// if a mouse click (touch) on the screen was detected...
//
Event_MouseLongClick(Gr,x,y)
begin
// stub to satisfy compiler
end;
//
// put here code you want to execute
// if a mouse click (touch) on the screen was detected...
//
Event_MouseClick(Gr,x,y)
begin
case
if (Gr==main_window) then
select_element(Gr,x,y,0);
end;

if (Gr==property_window) then
show_element(property_window,selected,y);
// DEBUG: textout_p(ip((PropWndBeg+y)/linH),Gr,1,ip((PropWndBeg+y)/linH)*linH,1);
end;

if (Gr==list_window) then
// we are on the header list
if (y<13) then
// sort list according to clicked header
case
// local xx:=30;
// local xxx:=70;
// local xxxx:=180;
if (0<x<=30) then
// sort numeric
LCurrSortOrder:=0;
end;
if (71<x<=180) then
// sort alphabetically
LCurrSortOrder:=1;
end;
end;
show_list(Gr,LCurrSortOrder,x,y);
else
// selected clicked element
show_list(Gr,LCurrSortOrder,x,y);
end;
end;

if (Gr==about_window) then
// DEBUG:
end;

if (Gr==plot_window) then
selected:=selected_plot(x,y);
show_plot(Gr,selected_prop,selected,connect_dots);
end;

end;
end;
//
//
//
Event_DragStart(Gr,x,y)
begin
```

```
// stub to satisfy compiler
end;
//
//
//
Event_DragEnd(Gr,x,y)
begin
// stub to satisfy compiler
end;
//
// Drag Move
//
Event_DragMove(Gr,x1,y1,x0,y0)
begin
case
if (Gr==main_window) then
MainOffset:=ip((x0-x1)/MainSmooth);
if (MainWndEnd+MainOffset)>HSIZE then
MainWndBeg:=HSIZE-320;
MainWndEnd:=HSIZE;
MainDOffset:=HSIZE-320-2;
else
if (MainWndBeg+MainOffset)<0 then
MainWndBeg:=0;
MainWndEnd:=320;
MainDOffset:=0
else
MainWndBeg:=MainWndBeg+MainOffset;
MainWndEnd:=MainWndEnd+MainOffset;
MainDOffset:=MainDOffset+MainOffset;
end;
end;
end;
if (Gr==property_window) then
PropOffset:=ip((y0-y1)/PropSmooth);
if abs(PropOffset)>0 then
if (PropWndEnd+PropOffset)>VSIZE then
PropWndBeg:=VSIZE-320;
PropWndEnd:=VSIZE;
PropDOffset:=VSIZE-320-2;
else
if (PropWndBeg+PropOffset)<0 then
PropWndBeg:=0;
PropWndEnd:=320;
PropDOffset:=0
else
PropWndBeg:=PropWndBeg+PropOffset;
PropWndEnd:=PropWndEnd+PropOffset;
PropDOffset:=PropDOffset+PropOffset;
end;
end;
else
local mov:=x1-x0;
if (abs(mov)>0) then
// move fast through elements
selected:=selected+ip(mov/2);
// boundaries
if (selected<1) then selected:=MAX_ELEMENTS; end;
if (selected>MAX_ELEMENTS) then selected:=1; end;
show_element(property_window,selected,0);
end;
end;
end;
if (Gr==list_window) then
ListOffset:=ip((y0-y1)/ListSmooth);
if (ListWndEnd+ListOffset)>LSIZE then
ListWndBeg:=LSIZE-320;
ListWndEnd:=LSIZE;
ListDOffset:=LSIZE-320-2;
else
if (ListWndBeg+ListOffset)<0 then
ListWndBeg:=0;
ListWndEnd:=320;
ListDOffset:=0
else
ListWndBeg:=ListWndBeg+ListOffset;
ListWndEnd:=ListWndEnd+ListOffset;
ListDOffset:=ListDOffset+ListOffset;
end;
end;
end;
if (Gr==about_window) then
// DEBUG:
end;
```

```
if (Gr==plot_window) then
// DEBUG:
end;
end;
end;
//
// Soft Menu Handler
//
Event_MenuClick(Gr,Index)
begin
case
if (Gr==main_window) then
case
if (Index==1) then // <List>
show_list(list_window,0,0,0);
DoLoop(list_window);
end;
if (Index==2) then // <Plot>
// prop_plot: select property
selected_prop:=prop_plot;
show_plot(plot_window,selected_prop,selected,connect_dots);
DoLoop(plot_window);
end;
if (Index==3) then // <>
end;
if (Index==4) then // <Help>
end;
if (Index==5) then // <About>
show_about(about_window);
DoLoop(about_window);
end;
if (Index==6) then //<Quit>
// trigger DoLoop to exit condition;
Event_KeyPress(Gr,K_ESC);
end; // if
end; // case
end;
if (Gr==plot_window) then
// if we are above the checkmark or the text we toggle the checkmark
// cheat as if it were menu 5 or menu 6
case
if (3<Index<6) then // Above Checkmark?
// toggle flag
if (connect_dots==0) then
connect_dots:=1
else
connect_dots:=0;
end;
show_plot(plot_window,selected_prop,selected,connect_dots);
end;
end;
end; // if
end; // case
end;
//
// Keyboard Handler
//
Event_KeyPress(Gr,KeyCode)
begin
case
// Main Window
if (Gr==main_window) then
case
if (KeyCode==K_ENTER) then
show_element(property_window,selected,0);
DoLoop(property_window);
end;
if (KeyCode==K_PLOT) then
// simply relay it to the menu click
Event_MenuClick(Gr,2)
end;
if (KeyCode==K_ARROWUP) then
// select the visually upper element in table
previous:=selected;
selected:=move_up_to(previous);
select_element(Gr,0,0,selected);
Z:=selected;
end;
if (KeyCode==K_ESC) then
// Escape is our exit code
// do not catch it here but in the DoLoop
break;
end;
if (KeyCode==K_ARROWLEFT) then
```

```
// select previous element in table
previous:=selected;
if previous==1 then
selected:=MAX_ELEMENTS;
else
selected:=selected-1;
end;
select_element(Gr,0,0,selected);
Z:=selected;
end;
if (KeyCode==K_ARROWRIGHT) then
// select the next element in table
previous:=selected;
if previous==MAX_ELEMENTS then
selected:=1;
else
selected:=selected+1;
end;
select_element(Gr,0,0,selected);
Z:=selected;
end;
if (KeyCode==K_ARROWDOWN) then
// select the visually lower element in table
previous:=selected;
selected:=move_down_to(previous);
select_element(Gr,0,0,selected);
Z:=selected;
end;
end;
end;
// Property Window
if (Gr==property_window) then
case
// Go back to the table view
if (KeyCode==K_HOME) then
// Go back to the table view
// (call us again this time with breaking the Message Loop)
Event_KeyPress(Gr,K_ESC)
end;
if (KeyCode==K_ARROWUP) then
PropWndBeg:=0;
PropWndEnd:=320;
PropDOffset:=0
end;
if (KeyCode==K_ARROWDOWN) then
PropWndBeg:=VSIZE-320;
PropWndEnd:=VSIZE;
PropDOffset:=VSIZE-320-2;
end;
if (KeyCode==K_ARROWLEFT) then
// Go one atom number lower or else underflow to '118 UUo' (MAX_ELEMENTS)
previous:=selected;
selected:=selected-1;
if (selected<1) then selected:=MAX_ELEMENTS; end;
show_element(property_window,selected,0);
end;
if (KeyCode==K_ARROWRIGHT) then
// Go one atom number higher or else overflow to '1 H' (1)
previous:=selected;
selected:=selected+1;
if (selected>MAX_ELEMENTS) then selected:=1; end;
show_element(property_window,selected,0);
end;
end;
end;
// List Window
if (Gr==list_window) then
case
// Go back to the table view
if (KeyCode==K_HOME) then
// Go back to the table view
// (call us again this time with breaking the Message Loop)
Event_KeyPress(Gr,K_ESC)
end;
if (KeyCode==K_ARROWUP) then
ListWndBeg:=0;
ListWndEnd:=320;
ListDOffset:=0
end;
if (KeyCode==K_ARROWDOWN) then
ListWndBeg:=LSIZE-320;
ListWndEnd:=LSIZE;
ListDOffset:=LSIZE-320-2;
end;
```

```
if (KeyCode==K_ENTER) then
if (0<selected<MAX_ELEMENTS) then
if (LCurrSortOrder==0) then
show_element(property_window,selected,0);
else
// if sorted alphabetically we need to lookup the number first
// we can no longer assume list position equals atom number
show_element(property_window,pos(elementname, new_order(selected)),0);
end;
DoLoop(property_window);
end;
end;
end;
end;
// Credit Window
if (Gr==about_window) then
case
// Go back to the table view
if (KeyCode==K_HOME) then
// Go back to the table view
// (call us again this time with breaking the Message Loop)
Event_KeyPress(Gr,K_ESC);
break;
end;
end;
end;
// Plot window
if (Gr==plot_window) then
case
// Go back to the table view
if (KeyCode==K_HOME) then
// Go back to the table view
// (call us again this time with breaking the Message Loop)
Event_KeyPress(Gr,K_ESC);
break;
end;
if (KeyCode==K_ARROWLEFT) then
// Go one atom number lower or else underflow to '118 UUo' (MAX_ELEMENTS)
previous:=selected;
selected:=selected-1;
if (selected<1) then selected:=MAX_ELEMENTS; end;
show_plot(Gr,selected_prop,selected,connect_dots);
end;
if (KeyCode==K_ARROWRIGHT) then
// Go one atom number higher or else overflow to '1 H' (1)
previous:=selected;
selected:=selected+1;
if (selected>MAX_ELEMENTS) then selected:=1; end;
show_plot(Gr,selected_prop,selected,connect_dots);
end;
end;
end;
end;
end;
//
// select the previous adjacent visually
//
move_up_to(selEl)
begin
case
if selEl==1 then return 87; end;
if selEl==2 then return 118; end;
if selEl==3 then return 1; end;
if selEl==4 then return 88; end;
if (selEl>=5 and selEl<10) then return selEl+94; end;
if (selEl>=10 and selEl<21) then return selEl-8; end;
if (selEl>=21 and selEl<31) then return selEl+68; end;
if (selEl>=58 and selEl<72) then return selEl+46; end;
if (selEl>=71 and selEl<=MAX_ELEMENTS) then return selEl-32; end;
default return selEl-18;
end;
end;
//
// select the adjacent visually
//
move_down_to(selEl)
begin
case
if selEl==1 then return 3; end;
if (selEl>=2 and selEl<13) then return selEl+8; end;
if (selEl>=40 and selEl<90) then return selEl+32; end;
if (selEl>=90 and selEl<99) then return selEl-68; end;
if (selEl>=99 and selEl<104) then return selEl-94; end;
if (selEl>=104 and selEl<MAX_ELEMENTS) then return selEl-46; end;
if (selEl=MAX_ELEMENTS) then return selEl-116; end;
default return selEl+18;
end;
end;
//
// draw a cross
//
cross_point(Gr,x,y,color)
begin
line_p(Gr,x,y-1,x,y-3,color);
line_p(Gr,x+1,y,x+3,y,color);
line_p(Gr,x,y+1,x,y+3,color);
line_p(Gr,x-1,y,x-3,y,color);
end;
//
// draw a cross
//
check_mark(Gr,x,y,color,on_off)
begin
line_p(Gr,x,y,x+8,y,color);
line_p(Gr,x+8,y,x+8,y+8,color);
line_p(Gr,x+8,y+8,x,y+8,color);
line_p(Gr,x,y+8,x,y,color);
if (on_off<>0) then
line_p(Gr,x,y+3,x+4,y+8,color);
line_p(Gr,x+4,y+8,x+8,y,color);
else
line_p(Gr,x,y+3,x+4,y+8,WHITE);
line_p(Gr,x+4,y+8,x+8,y,WHITE);
end;
end;
//
// draw an arrow down
//
down_arrow(Gr,x,y,color)
begin
line_p(Gr,x+3,y,x+7,y,color); //horizontal line
line_p(Gr,x+3,y,x+3,y+10,color); // first vertical line
line_p(Gr,x+7,y,x+7,y+10,color); // second vertical line
line_p(Gr,x,y+10,x+3,y+10,color); // left horz line arrow
line_p(Gr,x+7,y+10,x+10,y+10,color); // right horz line arrow
line_p(Gr,x,y+10, x+5,y+15,color); // left slope towards peak
line_p(Gr,x+10,y+10, x+5,y+15,color); // right slope towards peak
end;
//
// draw an arrow up
//
up_arrow(Gr,x,y,color)
begin
line_p(Gr,x+3,y+15,x+7,y+15,color); //horizontal line
line_p(Gr,x+3,y+15,x+3,y+5,color); // first vertical line
line_p(Gr,x+7,y+15,x+7,y+5,color); // second vertical line
line_p(Gr,x,y+5,x+3,y+5,color); // left horz line arrow
line_p(Gr,x+7,y+5,x+10,y+5,color); // right horz line arrow
line_p(Gr,x,y+5, x+5,y,color); // left slope towards peak
line_p(Gr,x+10,y+5, x+5,y,color); // right slope towards peak
end;
//
// left arrow
//
left_arrow(Gr,x,y,color)
begin
line_p(Gr,x+6,y,x+6,y+6,color); // vertical line
line_p(Gr,x,y+3,x+6,y,color); // sloped upper line
line_p(Gr,x,y+3,x+6,y+6,color); // sloped lower line
end;
//
// right arrow
//
right_arrow(Gr,x,y,color)
begin
line_p(Gr,x,y,x,y+6,color); // vertical line
line_p(Gr,x,y,x+6,y+3,color); // sloped upper line
line_p(Gr,x,y+6,x+6,y+3,color); // sloped lower line
end;
//
// get back random color
//
random_color()
begin
return rgb(random(256),random(256),random(256));
end;
//
// build the table
```

```
//
build_table(Gr)
begin
local i; //used by matrix
local j; // used by matrix
local x:=2; // MainOffset to start
local dx:=0; // offset relative to the elements abbreviation length
local y:=2; // MainOffset to start
local TablePos:=1;

dimgrob_p(Gr,HSIZE,220);
rect_p(Gr);
blit_p(G0,Gr,0,PropWndBeg,320,PropWndEnd);

for j:=1 to 10 do
for i:=1 to 18 do
if TablePos:=position(j,i) then
rect_p(Gr,x,y,x+el_width,y+el_height,rgb(0,0,0),distinct_colors(colorindex(TablePos)));
textout_p(TablePos,Gr,x+2,y+1,1,BLACK);
dx:=dim(symbolname(TablePos))-2;
textout_p(symbolname(TablePos),Gr,x+text_offset-dx,y+10,1,BLACK);
end;
x:=x+el_width;
end;
x:=x-18*el_width;
y:=y+el_height;
end;
x:=2*el_width+el_width/4+2; y:=7*el_height+2;

// title
textout_p(langreslist(LANG_ID+2),Gr,100+x-45,5,3,#FF006E);

// left columns
rect_p(Gr,51+x-45,26,55+x-45,30,distinct_colors(1));
textout_p(distinct_family(5),Gr,58+x-45,25,1); //"Nichtmetalle"

rect_p(Gr,51+x-45,36,55+x-45,40,distinct_colors(2));
textout_p(distinct_family(1),Gr,58+x-45,35,1); //"Edelgase"

rect_p(Gr,51+x-45,46,55+x-45,50,distinct_colors(3));
textout_p(distinct_family(2),Gr,58+x-45,45,1); //"Alkalimetalle"

rect_p(Gr,51+x-45,56,55+x-45,60,distinct_colors(4));
textout_p(distinct_family(3),Gr,58+x-45,55,1); //"Erdalkalinemetalle"
// right columns
rect_p(Gr,160+x-45,26,164+x-45,30,distinct_colors(5));
textout_p(distinct_family(8),Gr,168+x-45,25,1); //"Übergangsmetalle"

rect_p(Gr,160+x-45,36,164+x-45,40,distinct_colors(6));
textout_p(distinct_family(4),Gr,168+x-45,35,1); //"Halbmetalle"

rect_p(Gr,160+x-45,46,164+x-45,50,distinct_colors(7));
textout_p(distinct_family(7),Gr,168+x-45,45,1); //"Metalle"

rect_p(Gr,160+x-45,56,164+x-45,60,distinct_colors(8));
textout_p(distinct_family(6),Gr,168+x-45,55,1); //"Halogene"

// lower two series
rect_p(Gr,61+x-45,152,65+x-45,156,distinct_colors(9));
textout_p(distinct_family(9),Gr,68+x-45,151,1); //"Lanthaniden"

rect_p(Gr,61+x-45,161,65+x-45,165,distinct_colors(10));
textout_p(distinct_family(10),Gr,68+x-45,160,1); //"Actiniden"

// connection lines to lanthanides
line_p(Gr,x,y,x,y+2*el_height+el_height/2);
line_p(Gr,x,y+2*el_height+el_height/2,x+el_width*3/4,y+2*el_height+el_height/2);
// connection lines to actinides
x:=x+el_width/2;
line_p(Gr,x,y,x,y+el_height+el_height/2);
line_p(Gr,x,y+el_height+el_height/2,x+el_width/4,y+el_height+el_height/2);
end;
//
// invert the selected element
//
select_element(Gr,x,y,sel)
begin
local i; //used by matrix
local j; // used by matrix
local TablePos:=1;
local txt_off:=0;
local dx:=0;

if ((x+y)>0) then
```

```
// calculate based on the mouse position the
// granular element in the matrix
local tempX:=(x-2+MainDOffset)/el_width;
local tempY:=(y-2)/el_height;
// POSITION OF ELEMENT IN TABLE
selected:=position(ip(tempY)+1,ip(tempX)+1);
else
selected:=sel;
end;

x:=2;
y:=2;
for j:=1 to 10 do
for i:=1 to 18 do
if TablePos:=position(j,i) then
rect_p(Gr,x,y,x+el_width,y+el_height,BLACK,distinct_colors(colorindex(TablePos)));
dx:=dim(symbolname(TablePos))-2;
if selected==TablePos then
// wipe out (overwrite current title)
rect_p(Gr,30,2,260,22,WHITE,WHITE);
// write new title
textout_p(elementname(TablePos),Gr,30,5,3,#FF006E);
// right align text
if (selected<=4) then txt_off:=0; end;
if (selected>4 and selected<=43) then txt_off:=6; end;
if (selected>43) then txt_off:=12; end;
textout_p(round(molmasse(TablePos),0),Gr,150-txt_off,2,2,0);
// write symbolname
textout_p(symbolname(TablePos),Gr,160,5,3,#FF006E);
// right align text
if (selected<10) then txt_off:=0; end;
if (selected>=10 and selected<=99) then txt_off:=6; end;
if (selected>99) then txt_off:=12; end;
textout_p(TablePos,Gr,150-txt_off,12,2,0);
// invert selected element
fillpoly_p(Gr,{x,y,x,y+el_height,x+el_width,y+el_height,x+el_width,y},BLACK,110);
textout_p(TablePos,Gr,x+2,y+1,1,WHITE);
textout_p(symbolname(TablePos),Gr,x+text_offset-dx,y+10,1,WHITE);
else
textout_p(TablePos,G1,x+2,y+1,1,BLACK);
textout_p(symbolname(TablePos),Gr,x+text_offset-dx,y+10,1,BLACK);
end;
end;
x:=x+el_width;
end;
x:=x-18*el_width;
y:=y+el_height;
end;
end;
//
// Show the selected Element
//
show_element(Gr,sel,y)
begin
local EX1:=10,EX2:=130,EX3:=286;
local txt:="",txt1:="",txt2:="";
local BRK:=28;
local lin:=0;
local highlight_prop:=0;
local props:={}, props1,props2;
local j;
local n:=0;

// which property on which line?
props1:={0,0,17,14,32,30,31,28,0,0,3,20,21,22,7,25,0,0,0,8,18,5,9,10,19,11,12,13,24,23,0,0,6,27,26};
props2:={0,0,17,14,32,32,30,31,28,0,0,3,20,21,22,7,25,0,0,0,8,18,5,9,10,19,11,12,13,24,23,0,0,6,27,26};
props:=props1;
// determine list to use
if (dim(distinct_lookalike(lookalike(sel)))>BRK) then
props:=props2;
end;

// select & highlight correct property
if (y>=linH) then
lin:=ip((PropWndBeg+y)/linH);
highlight_prop:=props(lin);
end;

// create the composed graphic
dimgrob_p(property_window,320,VSIZE);
rect_p(property_window);

// Element [-], Col:2
textout_p(elementname(sel),Gr,7,2,3,rgb(255,0,110));
```

```
// Name [-], Col:1
textout_p(symbolname(sel),Gr,EX2+30,2,3,rgb(255,105,0));
// Atomic Number [-] Col:0 (implicit column)
textout_p(sel,Gr,EX2,2,3,rgb(38,127,0));

// If it is Technetium(43) or Promethium(61) or
// between Neptunium(93) and the biggest today known element
// then it is a syntetic element
if (sel==43 or sel==61 or (93≤sel and sel≤112)) then
textout_p("("+langreslist(LANG_ID+8)+")",Gr,EX2+60,7,1,rgb(255,0,0));
end;

// first page of properties

// General properties
textout_p(langreslist(LANG_ID+9),Gr,EX1,2*linH-3,2);

// Category, Family [-], Col:17
textout_p(headerlist(17)+":",Gr,EX1,(3)*linH,1,LIGHTBLUE);
if (highlight_prop==17) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(distinct_family(family(sel)),Gr,EX2,(3)*linH,1,WHITE);
else
textout_p(distinct_family(family(sel)),Gr,EX2,(3)*linH,1,BLACK);
end;

// Group, Period, Block [-], Col:14, Col:15, Col:16
textout_p(headerlist(14)+", "+headerlist(15)+", "+headerlist(16)+":",Gr,EX1,(4)*linH,1,LIGHTBLUE);
if (highlight_prop==14) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(group(sel)+", "+period(sel)+", "+block(sel),Gr,EX2,(4)*linH,1,WHITE);
else
textout_p(group(sel)+", "+period(sel)+", "+block(sel),Gr,EX2,(4)*linH,1,BLACK);
end;

// Look Alike [-], Col:32 -- optional second line of text
textout_p(headerlist(32)+":",Gr,EX1,(5)*linH,1,LIGHTBLUE);
txt1:=distinct_lookalike(lookalike(sel));
if (dim(txt1)>BRK) then
// find the closest space from string backwards
// starting with char(28) 'BRK'
txt2:=left(txt1,BRK);
for j from BRK downto 1 do
txt:=mid(txt2,j,1);
if txt==" " then break; end;
end;
txt2:=right(txt1,dim(txt1)-j);
txt1:=left(txt1,j-1);
n:=1;
end;

if (highlight_prop==32) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(txt1,Gr,EX2,(5)*linH,1,WHITE);
if (dim(txt2)>0) then
if (lin==6) then
// if we selected the second line select the previous too (it seems that if you would have selected both
lines)
fillpoly_p(Gr,{EX2-1,(lin+0-n)*linH-2,EX2-1,(lin+1-n)*linH-3,EX3,(lin+1-n)*linH-3,EX3,(lin+0-n)*linH-
2},BLACK,110);
textout_p(txt1,Gr,EX2,(5)*linH,1,WHITE);
else
// select also the next line (it seems that if you would have selected both lines)
fillpoly_p(Gr,{EX2-1,(lin+0+n)*linH-2,EX2-1,(lin+1+n)*linH-3,EX3,(lin+1+n)*linH-3,EX3,(lin+0+n)*linH-
2},BLACK,110);
end;
textout_p(txt2,Gr,EX2,(6)*linH,1,WHITE);
end;
else
textout_p(txt1,Gr,EX2,(5)*linH,1,BLACK);
if (dim(txt2)>0) then
textout_p(txt2,Gr,EX2,(6)*linH,1,BLACK);
end;
end;

// CAS Number [-], Col:30
textout_p(headerlist(30)+":",Gr,EX1,(6+n)*linH,1,LIGHTBLUE);
if (highlight_prop==30) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(CASnumber(sel),Gr,EX2,(6+n)*linH,1,WHITE);
else
textout_p(CASnumber(sel),Gr,EX2,(6+n)*linH,1,BLACK);
end;
```

```
// Mass on Earth Shell [ppm], Col:31
textout_p(headerlist(31)+":",Gr,EX1,(7+n)*linH,1,LIGHTBLUE);
if (highlight_prop==31) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(string(earthmass(sel),3,2)+" "+unitlist(31),Gr,EX2,(7+n)*linH,1,WHITE);
else
textout_p(string(earthmass(sel),3,2)+" "+unitlist(31),Gr,EX2,(7+n)*linH,1,BLACK);
end;

// Discovery Year [Year],Col:28
textout_p(headerlist(28)+":",Gr,EX1,(8+n)*linH,1,LIGHTBLUE);
if (highlight_prop==28) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(discovery(sel)>0,discovery(sel)+"
"+unitlist(28),langreslist(LANG_ID+10)),Gr,EX2,(8+n)*linH,1,WHITE);
else
textout_p(ifte(discovery(sel)>0,discovery(sel)+"
"+unitlist(28),langreslist(LANG_ID+10)),Gr,EX2,(8+n)*linH,1,BLACK);
end;

// Atomic properties
textout_p(langreslist(LANG_ID+11),Gr,10,(10+n)*linH-3,2);

// Atomic Weight [u], Col:3
textout_p(headerlist(3)+":",Gr,EX1,(11+n)*linH,1,LIGHTBLUE);
if (highlight_prop==3) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(atomicmass(sel)+" "+unitlist(3),Gr,EX2,(11+n)*linH,1,WHITE);
else
textout_p(atomicmass(sel)+" "+unitlist(3),Gr,EX2,(11+n)*linH,1,BLACK);
end;

// Atomic Radius [pm], Col:20
textout_p(headerlist(20)+":",Gr,EX1,(12+n)*linH,1,LIGHTBLUE);
if (highlight_prop==20) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(atomradius(sel)>0,atomradius(sel)+" "+unitlist(20),"-"),Gr,EX2,(12+n)*linH,1,WHITE);
else
textout_p(ifte(atomradius(sel)>0,atomradius(sel)+" "+unitlist(20),"-"),Gr,EX2,(12+n)*linH,1,BLACK);
end;
// Covalent radius [pm], Col:21
textout_p(headerlist(21)+":",Gr,EX1,(13+n)*linH,1,LIGHTBLUE);
if (highlight_prop==21) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(covalentradii(sel)>0,covalentradii(sel)+" "+unitlist(21),"-"),Gr,EX2,(13+n)*linH,1,WHITE);
else
textout_p(ifte(covalentradii(sel)>0,covalentradii(sel)+" "+unitlist(21),"-"),Gr,EX2,(13+n)*linH,1,BLACK);
end;

// Van-der-Waals-Radius [pm], Col:22
textout_p(headerlist(22)+":",Gr,EX1,(14+n)*linH,1,LIGHTBLUE);
if (highlight_prop==22) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(vanderwaals(sel)>0,vanderwaals(sel)+" "+unitlist(22),"-"),Gr,EX2,(14+n)*linH,1,WHITE);
else
textout_p(ifte(vanderwaals(sel)>0,vanderwaals(sel)+" "+unitlist(22),"-"),Gr,EX2,(14+n)*linH,1,BLACK);
end;

// Electron Configuration [-], Col:7
textout_p(headerlist(7)+":",Gr,EX1,(15+n)*linH,1,LIGHTBLUE);
if (highlight_prop==7) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(elektronconfig(sel),Gr,EX2,(15+n)*linH,1,WHITE);
else
textout_p(elektronconfig(sel),Gr,EX2,(15+n)*linH,1,BLACK);
end;

// First Ionisation Energy [kJ/mol], Col:25
textout_p(headerlist(25)+":",Gr,EX1,(16+n)*linH,1,LIGHTBLUE);
if (highlight_prop==25) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(firstionenergy(sel)>0,firstionenergy(sel)+" "+unitlist(25),"-"),Gr,EX2,(16+n)*linH,1,WHITE);
else
textout_p(ifte(firstionenergy(sel)>0,firstionenergy(sel)+" "+unitlist(25),"-"),Gr,EX2,(16+n)*linH,1,BLACK);
end;
//second page of properties (does not need to react to text moves above ('look alike' prop))
// determine list to use
props:=props1;
// select & highlight correct property
if (y>=linH) then
lin:=ip((PropWndBeg+y)/linH);
highlight_prop:=props(lin);
end;
n:=0;
```

```
// Name [-], Col:1
textout_p(symbolname(sel),Gr,EX2+30,(19)*linH-6,3,rgb(255,105,0));
// Atomic Number [-] Col:0 (implicit column)
textout_p(sel,Gr,EX2,(19)*linH-6,3,rgb(38,127,0));

// Physical Properties
textout_p(langreslist(LANG_ID+12),Gr,10,(19)*linH-3,2,0);

// Phase (Aggregate State at Room Temp) [-], Col:8
textout_p(headerlist(8)+":",Gr,EX1,(20+n)*linH,1,LIGHTBLUE);
if (highlight_prop==8) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(distinct_aggregatstate(aggregatstate(sel)),Gr,EX2,(20+n)*linH,1,WHITE);
else
textout_p(distinct_aggregatstate(aggregatstate(sel)),Gr,EX2,(20+n)*linH,1,BLACK);
end;

// Crystal Structure [-], Col:18
textout_p(headerlist(18)+":",Gr,EX1,(21+n)*linH,1,LIGHTBLUE);
if (highlight_prop==18) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(distinct_crystal(crystal(sel)),Gr,EX2,(21+n)*linH,1,WHITE);
else
textout_p(distinct_crystal(crystal(sel)),Gr,EX2,(21+n)*linH,1,BLACK);
end;

// Density [g/cm^3], Col:5
textout_p(headerlist(5)+":",Gr,EX1,(22+n)*linH,1,LIGHTBLUE);
if (highlight_prop==5) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(density(sel)>0,density(sel)+" "+unitlist(5),"-"),Gr,EX2,(22+n)*linH,1,WHITE);
else
textout_p(ifte(density(sel)>0,density(sel)+" "+unitlist(5),"-"),Gr,EX2,(22+n)*linH,1,BLACK);
end;

// Melting Point [°C], Col:9
textout_p(headerlist(9)+":",Gr,EX1,(23+n)*linH,1,LIGHTBLUE);
if (highlight_prop==9) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(meltingpoint(sel)>0,meltingpoint(sel)+" "+unitlist(9),"-"),Gr,EX2,(23+n)*linH,1,WHITE);
else
textout_p(ifte(meltingpoint(sel)>0,meltingpoint(sel)+" "+unitlist(9),"-"),Gr,EX2,(23+n)*linH,1,BLACK);
end;

// Boiling Point [°C], Col:10
textout_p(headerlist(10)+":",Gr,EX1,(24+n)*linH,1,LIGHTBLUE);
if (highlight_prop==10) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(boilingpoint(sel)>0,boilingpoint(sel)+" "+unitlist(10),"-"),Gr,EX2,(24+n)*linH,1,WHITE);
else
textout_p(ifte(boilingpoint(sel)>0,boilingpoint(sel)+" "+unitlist(10),"-"),Gr,EX2,(24+n)*linH,1,BLACK);
end;

// Atomic Volume [cm^3/mol], Col:19
textout_p(headerlist(19)+":",Gr,EX1,(25+n)*linH,1,LIGHTBLUE);
if (highlight_prop==19) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(atomicvolume(sel)>0,atomicvolume(sel)+" "+unitlist(19),"-"),Gr,EX2,(25+n)*linH,1,WHITE);
else
textout_p(ifte(atomicvolume(sel)>0,atomicvolume(sel)+" "+unitlist(19),"-"),Gr,EX2,(25+n)*linH,1,BLACK);
end;

// Heat of Vaporization [kJ/mol], Col:11 // Boiling Enthalpy
textout_p(headerlist(11)+":",Gr,EX1,(26+n)*linH,1,LIGHTBLUE);
if (highlight_prop==11) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(boilingenthalpy(sel)>0,boilingenthalpy(sel)+" "+unitlist(11),"-"),Gr,EX2,(26+n)*linH,1,WHITE);
else
textout_p(ifte(boilingenthalpy(sel)>0,boilingenthalpy(sel)+" "+unitlist(11),"-"),Gr,EX2,(26+n)*linH,1,BLACK);
end;

// Heat of Fusion [kJ/mol], Col:12 // Melting Enthalpy
textout_p(headerlist(12)+":",Gr,EX1,(27+n)*linH,1,LIGHTBLUE);
if (highlight_prop==12) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(meltingenthalpy(sel)>0,meltingenthalpy(sel)+" "+unitlist(12),"-"),Gr,EX2,(27+n)*linH,1,WHITE);
else
textout_p(ifte(meltingenthalpy(sel)>0,meltingenthalpy(sel)+" "+unitlist(12),"-"),Gr,EX2,(27+n)*linH,1,BLACK);
end;

// Heat Capacity [J/(kg.K)], Col:13
textout_p(headerlist(13)+":",Gr,EX1,(28+n)*linH,1,LIGHTBLUE);
```

```
if (highlight_prop==13) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(heatcapacity(sel)>0,heatcapacity(sel)+" "+unitlist(13)),Gr,EX2,(28+n)*linH,1,WHITE);
else
textout_p(ifte(heatcapacity(sel)>0,heatcapacity(sel)+" "+unitlist(13)),Gr,EX2,(28+n)*linH,1,BLACK);
end;

// Electrical Conductivity [10^6 A/(V*m)], Col:24
textout_p(headerlist(24)+":",Gr,EX1,(29+n)*linH,1,LIGHTBLUE);
if (highlight_prop==24) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(electroconductivity(sel)>0,electroconductivity(sel)+" "+unitlist(24),"-
"),Gr,EX2,(29+n)*linH,1,WHITE);
else
textout_p(ifte(electroconductivity(sel)>0,electroconductivity(sel)+" "+unitlist(24),"-
"),Gr,EX2,(29+n)*linH,1,BLACK);
end;

// Thermal Conductivity [W/(m*K)], Col:23
textout_p(headerlist(23)+":",Gr,EX1,(30+n)*linH,1,LIGHTBLUE);
if (highlight_prop==23) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(ifte(thermalconductivity(sel)>0,thermalconductivity(sel)+" "+unitlist(23),"-
"),Gr,EX2,(30+n)*linH,1,WHITE);
else
textout_p(ifte(thermalconductivity(sel)>0,thermalconductivity(sel)+" "+unitlist(23),"-
"),Gr,EX2,(30+n)*linH,1,BLACK);
end;

// Chemical properties
textout_p(langreslist(LANG_ID+13),Gr,10,(32+n)*linH-3,2,0);

// Oxidation States [-], Col:6
textout_p(headerlist(6)+":",Gr,EX1,(33+n)*linH,1,LIGHTBLUE);
if (highlight_prop==6) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(oxidstates(sel),Gr,EX2,(33+n)*linH,1,WHITE);
else
textout_p(oxidstates(sel),Gr,EX2,(33+n)*linH,1,BLACK);
end;

// Oxide [-], Col:27
textout_p(headerlist(27)+":",Gr,EX1,(34+n)*linH,1,LIGHTBLUE);
if (highlight_prop==27) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(distinct_oxide(oxide(sel)),Gr,EX2,(34+n)*linH,1,WHITE);
else
textout_p(distinct_oxide(oxide(sel)),Gr,EX2,(34+n)*linH,1,BLACK);
end;

// Elektronegativity [-], Col:26
textout_p(headerlist(26)+":",Gr,EX1,(35+n)*linH,1,LIGHTBLUE);
if (highlight_prop==26) then
fillpoly_p(Gr,{EX2-1,(lin+0)*linH-2,EX2-1,(lin+1)*linH-3,EX3,(lin+1)*linH-3,EX3,(lin+0)*linH-2},BLACK,110);
textout_p(elektronegativity(sel),Gr,EX2,(35+n)*linH,1,WHITE);
else
textout_p(elektronegativity(sel),Gr,EX2,(35+n)*linH,1,BLACK);
end;

// Arrows
if (PropWndBeg<VSIZE-320) then
up_arrow(Gr,308,19*linH,BLACK);
textout_p(langreslist(LANG_ID+14),Gr,275,21*linH,1,rgb(0,0,240));
textout_p(langreslist(LANG_ID+16),Gr,268,22*linH,1,rgb(0,0,240));
end;
if (PropWndEnd<VSIZE) then
down_arrow(Gr,308,14*linH,BLACK);
textout_p(langreslist(LANG_ID+15),Gr,263,16*linH,1,rgb(0,0,240));
textout_p(langreslist(LANG_ID+16),Gr,268,17*linH,1,rgb(0,0,240));
end;
// show indications
left_arrow(Gr,0,119,BLACK);
right_arrow(Gr,313,119,BLACK);
end;
//
// Show sorted list of Elements
//
show_list(Gr, sort_alpha,posx,posy)
begin
local j,k;
local x:=2,y:=2;
local fore_color;
local atom_no;
local xx:=30,xxx:=70,xxxx:=180;
```

```
// at what element are we currently
local sel:=ip((ListWndBeg+posy)/linH);
// update global, eventually used by prop window
selected:=sel;
// create the view
dimgrob_p(Gr,320,LSIZE);
rect_p(Gr);

// print the column headers
if (sort_alpha>0) then
textout_p("No" ,Gr,x,y+j*linH,1,0);
else
textout_p("No" ,Gr,x,y+j*linH,1,0);
textout_p(" ҉" ,Gr,x+10,y+j*linH,1,rgb(255,0,0));
end;

// vertical line
line_p(Gr,xx-2,y,xx-2,linH-2,0);
line_p(Gr,xx-2,1*linH+2,xx-2,LSIZE,LIGHTGRAY);

textout_p(headerlist(1),Gr,xx,y+j*linH,1,0);
// vertical line
line_p(Gr,xxx-2,y,xxx-2,linH-2,0);
line_p(Gr,xxx-2,1*linH+2,xxx-2,LSIZE,LIGHTGRAY);

if (sort_alpha>0) then
textout_p(headerlist(2),Gr,xxx,y+j*linH,1,0);
textout_p("҉ ",Gr,xxx+40,y+j*linH,1,rgb(255,0,0));
else
textout_p(headerlist(2),Gr,xxx,y+j*linH,1,0);
end;

// vertical line
line_p(Gr,xxxx-2,y,xxxx-2,linH-2,0);
line_p(Gr,xxxx-2,1*linH+2,xxxx-2,LSIZE,LIGHTGRAY);

textout_p(headerlist(17),Gr,xxxx,y+j*linH,1,0);
// horizontal line
line_p(Gr,x,1*linH,318,1*linH,0);
// vertical lines
if sort_alpha then
new_order:=sort(elementname); // sort list
atom_no:=makelist(X,X,1,MAX_ELEMENTS,1);
// print out
for j:=1 to MAX_ELEMENTS do
if j==sel then
fillpoly_p(Gr,{1,sel*linH,318,sel*linH,318,(sel+1)*linH,1,(sel+1)*linH},BLACK,110);
fore_color:=WHITE;
else
fore_color:=BLACK;
end;
textout_p(pos(elementname, new_order(j)),Gr,x,y+j*linH,1,fore_color);
textout_p(symbolname(pos(elementname, new_order(j))),Gr,xx,y+j*linH,1,fore_color);
textout_p(new_order(j),Gr,xxx,y+j*linH,1,fore_color);
textout_p(distinct_family(family(pos(elementname, new_order(j)))),Gr,xxxx,y+j*linH,1,fore_color); end;
else
// print out
for j:=1 to MAX_ELEMENTS do
if j==sel then
fillpoly_p(Gr,{1,sel*linH,318,sel*linH,318,(sel+1)*linH,1,(sel+1)*linH},BLACK,110);
fore_color:=WHITE;
else
fore_color:=BLACK;
end;
//distinct_colors(colorindex(j));
textout_p(j,Gr,x,y+j*linH,1,fore_color);
textout_p(symbolname(j),Gr,xx,y+j*linH,1,fore_color);
textout_p(elementname(j),Gr,xxx,y+j*linH,1,fore_color);
textout_p(distinct_family(family(j)),Gr,xxxx,y+j*linH,1,fore_color);
end;
end;
end;
//
// Get the calculated Text Width in pixel for a specified string
// does only subset from 65 to 122 (if char is outside return five as
// a rough median
//
GetTextExtent(txt,fsize)
begin
//
// this is Pixels/per char (for the small size font =1)
// counted across the entire screen width (320)
// devided by the count of chars between ~40 and ~106(I)
// sometimes cutting in half or some
// not taking into account kerning of adjacent different chars
// (if this happens)
// also not taking into account special chars
//
// representing character codes 65 to 90
// then 6 dummies and again 97 to 122
// "A":=6.013773584906 "a":=5.02345327
// "B":=6.013773584906 "b":=6.03773584906
// "C":=6.013773584906 "c":=5.000000000000
// "D":=7.03381642512 "d":=6.03503584906
// "E":=4.9982 "e":=5.05291052913
// "F":=4.9982 "f":=3.01886792453
// "G":=7.03381642512 "g":=5.05291052913
// "H":=7.03381642512 "h":=6.03503584906
// "I":=3.01886792453 "i":=3.01886792453
// "J":=5.000000000000 "j":=3.01886792453
// "K":=6.013773584906 "k":=5.05291052913
// "L":=5.000000000000 "l":=3.01886792453
// "M":=9.015873015875 "m":=9.015873015875
// "N":=7.03381642512 "n":=6.03773584906
// "O":=7.03381642512 "o":=6.03773584906
// "P":=6.03773584906 "p":=6.03773584906
// "Q":=7.03381642512 "q":=6.03773584906
// "R":=6.03773584906 "r":=4.02531955695
// "S":=4.9982 "s":=4.9982
// "T":=5.05291052913 "t":=3.01886792453
// "U":=7.03381642512 "u":=6.03773584906
// "V":=6.03773584906 "v":=4.9982
// "W":=9.015873015875 "w":=7.03381642512
// "X":=5.05291052913 "x":=5.05291052913
// "Y":=5.05291052913 "y":=5.05291052913
// "Z":=5.05291052913 "z":=4.02531955695
//
local
CharSet:={6.013773584906,6.013773584906,6.013773584906,7.03381642512,4.9982,4.9982,7.03381642512,7.0338164251
2,3.01886792453,5.000000000000,9.015873015875,7.03381642512,7.03381642512,6.03773584906,7.03381642512,6.03773
584906,4.9982,5.05291052913,7.03381642512,6.03773584906,9.015873015875,5.05291052913,5.05291052913,5.05291052
913,5.0,5.0,5.0,5.0,5.0,5.0,5.02345327,6.03773584906,5.000000000000,6.03503584906,5.05291052913,3.01886792453
,5.05291052913,6.03503584906,3.01886792453,3.01886792453,9.015873015875,6.03773584906,6.03773584906,6.0377358
4906,6.03773584906,4.02531955695,4.9982,7.03381642512,5.05291052913,5.05291052913,4.02531955695};

local OFFSET:=64; // A starts at 65
local i,sum,ntxt,nval;

for i from 1 to dim(txt) do
ntxt:=asc(mid(txt,i,1));
nval:=5.0*fsize;
if 64<ntxt<=122 then
nval=CharSet(OFFSET+i)*fsize;
end;
sum:=sum+nval;
end;
return sum;
end;
//
// 4.12 as a 'educated guess'
// try and error test of this sample txt median value of charwidth
//
center_text(txt,Gr,x_width,y)
begin
local wtxt:=GetTextExtent(txt,1);
textout_p(txt,Gr,(x_width-wtxt)/2,y,1,BLACK,320,WHITE);
end;
//
// right align text
//
ralign_text(txt,Gr,x,y)
begin
local wtxt:=GetTextExtent(txt,1);
textout_p(txt,Gr,x-wtxt,y,1,BLACK);
end;
//
// binary tree
//
chaos(Gr)
begin
local aa,mm,xx;

mm:=3.001;
repeat aa:=0.05;
repeat xx:=aa;
for I from 1 to 3 do
xx:=mm*xx*(1-xx);
end;
```

```
pixon_p(Gr,(mm-3)*320,xx*240,#0000FF);
aa:=aa+0.03;
until aa>0.5;
mm:=mm+0.003;
until mm>4;
end;
//
// Show Credits where credit is due...
//
show_about(Gr)
begin
local i;
// create the view
dimgrob_p(Gr,320,240);
rect_p(Gr);
chaos(Gr);
for i from 1 to (size(Credit)-1) do
center_text(Credit(i),Gr,320,2+i*linH);
end;
textout_p(Credit(size(Credit)),Gr,1,231,1,BLACK,#000F0F,WHITE);
end;
//
// customized plot function
//
show_plot(Gr,prop,sel,conn_dots)
begin
local i;
local ymin;
local ymax;
local ftor;
local EX1:=22,EX2:=76;
local wtxt:=0;
local neg_off;
local lastx, lasty;
local this_prop;
local res:=0;

// create the view
dimgrob_p(Gr,320,240);
rect_p(Gr);
// horizontal (we need two times 118pixels horizontal)
line_p(Gr,10,210,69,210,BLACK);
line_p(Gr,70,210,310,210,LIGHTGRAY);

for i from 70 to 310 step 10 do
// after 50 and 100 (remember 2*118 pixels, to use available screen space)
line_p(Gr,i,211,i,ifte((i-70) mod 100,213,214),BLACK);
line_p(Gr,i,211,i,ifte((i-70) mod 200,213,215),BLACK);
end;

textout_p("0",Gr,60,220,1,BLACK);
textout_p("50",Gr,165,220,1,BLACK);
textout_p("100",Gr,262,220,1,BLACK);
textout_p("[Element]",Gr,278,230,1,BLACK);

// vertical (we need to normalize values the from 0 to 200pixels)
line_p(Gr,70,200,70,220,BLACK);
line_p(Gr,70,10,70,210,LIGHTGRAY);

// plot title
textout_p("["+unitlist(prop)+"]",Gr,75,2,1,BLACK);
textout_p(headerlist(prop)+":",Gr,120,2,1,BLACK);
wtxt:=GetTextExtent(headerlist(prop)+":",1)+5;

// draw checkmark
check_mark(Gr,180,230,BLACK,1);
textout_p(langreslist(LANG_ID+37),Gr,190,230,1,BLACK);

// which property we need to plot?
case
if (prop==4) then this_prop:=molmasse; end;
if (prop==5) then this_prop:=density; end;
if (prop==9) then this_prop:=meltingpoint; end;
if (prop==10) then this_prop:=boilingpoint; end;
if (prop==11) then this_prop:=boilingenthalpy; end;
if (prop==12) then this_prop:=meltingenthalpy; end;
if (prop==13) then this_prop:=heatcapacity; end;
if (prop==19) then this_prop:=atomicvolume; end;
if (prop==20) then this_prop:=atomradius; end;
if (prop==21) then this_prop:=covalentradii; end;
if (prop==22) then this_prop:=vanderwaals; end;
if (prop==23) then this_prop:=thermalconductivity; end;
if (prop==24) then this_prop:=electroconductivity; end;
if (prop==25) then this_prop:=firstionenergy; end;
if (prop==26) then this_prop:=elektronegativity; end;
if (prop==28) then this_prop:=discovery; end;
if (prop==31) then this_prop:=earthmass; end;
default return 0;
end;

ymin:=min(this_prop);
ymax:=max(this_prop);
textout_p(this_prop(sel)+unitlist(prop),Gr,120+wtxt,2,1,#0000FF);

ftor:=200/(ymax-ymin); // this factor is how many specific units per pixel
neg_off:=ftor*ymin; // negativ or positiv offset of zero [in pixels] (avoid divide by zero later on)
if (neg_off<>0) then // paint a faint line where 0 would be (if we are not already at zero)
line_p(Gr,67,210+neg_off,310,210+neg_off,LIGHTGRAY);
textout_p("0",Gr,60,210+neg_off-3,1,LIGHTGRAY); // show where zero is
end;
// need to paint the connected line?
check_mark(Gr,180,230,0,conn_dots);
for i from 1 to MAX_ELEMENTS step 1 do
// paint always two pixels per element on the coordinate plot
// normalize the y-axis
if (conn_dots>0) then
// we should not paint starting from 0,0 (only inside coordinate window)
if ((lastx+lasty)>0) then
// connect the previous with this element
line_p(Gr,lastx,lasty,70+i+i,210-(ftor*this_prop(i)-neg_off),LIGHTGRAY);
end;
end;
// starting at relative left Y-Axis (70)
// starting the Y-Value from below at relative zero (210pix)
// in the direction of going up
// value the Y-Min if there is an offset (not startig at zero)
pixon_p(Gr,70+i+i,210-(ftor*this_prop(i)-neg_off),#0000FF);
pixon_p(Gr,71+i+i,210-(ftor*this_prop(i)-neg_off),#0000FF);
lastx:=70+i+i;
lasty:=210-(ftor*this_prop(i)-neg_off);
end;
// show the cross where we are right now
cross_point(Gr,71+sel+sel,210-(ftor*this_prop(sel)-neg_off),#FF0000);
// normalize y-axis lineal to 200
for i from 0 to 200 step 5 do
// after 50 and 100 (remember 2*118 pixels, to use available screen space)
line_p(Gr,ifte(i mod 50,67,60),10+i,69,10+i,BLACK);
end;
//
// normalize y-axis to 200
// NOTE: There is no relative scale here, the Y-max scale is always ymax
// in other words: if the ymin value is not zero
// the scale has to be squeezd with the offset percentage
// and not just shifted, the screen will not magically enlarge
//ftor:=(ymax-ymin)/200;
ftor:=ymax/200;
for i from 0 to 200 step 50 do
// after 50 and 100 (remember 2*118 pixels, to use available screen space)
if (i==0) then
// avoid divide by zero
ralign_text(string(round(ymin,1)),Gr,52,200);
else
ralign_text(string(round(ftor*i,1)),Gr,50,206-i);
end;
end;

// Name [-], Col:1
textout_p(symbolname(sel),Gr,EX1,220,1,rgb(255,105,0));
// Atomic Number [-] Col:0 (implicit column)
textout_p(sel,Gr,2,215,1,rgb(38,127,0));
// Atomic Number [-] Col:0 (implicit column)
textout_p(round(molmasse(sel),0),Gr,2,225,1,rgb(38,127,0));
// Element [-], Col:2
textout_p(elementname(sel),Gr,EX2,220,1,rgb(255,0,110));
end;
//
// which property to plot
//
prop_plot()
begin
// we are not able to plot non-numerical properties
//we should be able to plot those:
// 4: Mol Weight
// 5: Density
// 9: Melting Point
// 10: Boiling Point
// 11: Heat of Vaporization
// 12: Heat of Fusion
// 13: Heat Capacity
// 19: Atomic Volume
// 20: Atom Radius
// 21: Covalent Radius
// 22: Van-der-Waals-Radius
// 23: Thermal Conductivity
// 24: Electrical Conductivity
// 25: First Ionisation Energy
// 26: Elektronegativity
// 28: Discovery Year
// 31: Mass on Earth Shell
local prop_sel;
local props:={"4: "+headerlist(4), "5: "+headerlist(5), "6: "+headerlist(6), "7: "+headerlist(7), "8: 
"+headerlist(8),
"9: "+headerlist(9), "10: "+headerlist(10), "11: "+headerlist(11), "12: "+headerlist(12), "13: 
"+headerlist(13),
"14: "+headerlist(14), "15: "+headerlist(15), "16: "+headerlist(16), "17: "+headerlist(17), "18: 
"+headerlist(18),
"19: "+headerlist(19), "20: "+headerlist(20), "21: "+headerlist(21), "22: "+headerlist(22), "23: 
"+headerlist(23),
"24: "+headerlist(24), "25: "+headerlist(25), "26: "+headerlist(26), "27: "+headerlist(27), "28: 
"+headerlist(28),
"31: "+headerlist(31)};

choose (prop_sel, langreslist(LANG_ID+36),props);
// we start with the fourth prop and the last one is after 28th the 31st
return ifte(prop_sel==26,prop_sel+5,prop_sel+3);
end;
//
// which elemt are we at?
//
selected_plot(x,y)
begin
// first of all, we must be inside our defined Y-Axis of 200 pixels
if (10<=y<=210) then
// then wee need to inside 2*MAX_ELEMENTS
// we know that we start a X-Axis 70 pixel
if (70<x<70+2*MAX_ELEMENTS) then
return ip((x-70)/2);
end;
end;
end;
//
// core loop of the application
// loop until 'ESC' key has been detected
//
DoLoop(Gr)
begin
local MyWait, LastWait;
local MyMouse, LastMouse;
local MyKey, LastKey;
local WaitReal;

repeat // until battery is depleted completly
// this must be the first event to fire
// in order to properly show the new screen
// logically it could also be at the end (where it was first)
// but this gives a strange behaviour (user experience)
Event_Redraw(Gr);
// if it is a list save away the last event code
if type(MyWait)==6 then
LastWait:=MyWait;
end;
if type(MyMouse)==6 then
LastMouse:=MyMouse;
end;
// read from keyboard input queue
MyWait:=B→R(wait(-1));
// read from touch input queue
MyMouse:=B→R(mouse());
// if it is indeed a real number then a keystroke has been recorded
if (type(MyWait)==0) then
// store the keycode for later use
WaitReal:=MyWait;
if (MyWait>=0) and (MyWait<51) then
Event_KeyPress(Gr,MyWait);
end;
end; // if
// a list? (if it is a list then a mouse event occured)
if (type(MyWait)==6) then
// if it is a list, then a mouse (touch) has been recognized
case
if (MyWait(1)==0) then
// you may also check MyWait(1) for x1 and MyWait(2) for y1;
```

```
if (size(MyMouse(1))==5) and (size(MyMouse(2))==0) then // we expect a list as {x1,y1,x0,y0,type}
case
if (MyMouse(1,5)==0) then
Event_MouseDown(Gr,MyMouse(1,1),MyMouse(1,2));
end; // if
end; // case
end; // if
end; // if
if (MyWait(1)==1) then
case
if (size(MyMouse(1))==5) and (size(MyMouse(2))==0) then
if (MyMouse(1,5)==2) and (LastWait(1)=0) then
Event_DragStart(Gr,LastMouse(1,1),LastMouse(1,2));
end;
if (MyMouse(1,5)==2) and (LastWait(1)=1) then
Event_DragMove(Gr,MyMouse(1,1),MyMouse(1,2),LastMouse(1,1),LastMouse(1,2));
end; // if
end; // if
end; // case
end; // if
if (MyWait(1)==2) then
case
if (size(MyMouse(1))==0) and (size(MyMouse(2))==0) and (size(MyWait)==1) then
// {2} {{}{}}
case
if (LastWait(1)==0) then
Event_MouseClick(Gr,LastWait(2),LastWait(3));
end;
if (LastWait(1)==1) and (size(LastWait)==3) then
// {1,x,y}
if (size(LastMouse(1))==5) then
// CAUTION: you could put an AND instead of two ifs here
// but thats just logically so, if size() is turned out to be empty
// the next call would fail,
if (LastMouse(1,5)==2) then
// {{x1,y1,x0,y0,2}{}}
Event_DragEnd(Gr,LastMouse(1,1),LastMouse(1,2));
end;
end; // if
end; // if
if (LastWait(1)==7) and (size(LastWait)==3) then
// {7,x,y} {{}{}}
end; // if
end; // case
end; // if
end; // case
end; // if
if (MyWait(1)==3) then
if (MyWait(3)>= 213 and MyWait(3)<=239) then
// SOFTBUTTONS
// this is to check if we are down below on the softmenu area
case
// 1#6 menu points
if (0 <= MyWait(2) <= 51) then
Event_MenuClick(Gr,1);
end;
// 2#6 menu points
if (53 <= MyWait(2) <= 104) then
Event_MenuClick(Gr,2);
end;
// 3#6 menu points
if (106 <= MyWait(2) <=157) then
Event_MenuClick(Gr,3);
end;
// 4#6 menu points
if (159 <= MyWait(2) <=210) then
Event_MenuClick(Gr,4);
end;
// 5#6 menu points
if (212 <= MyWait(2) <=263) then
Event_MenuClick(Gr,5);
end;
// 6#6 menu points
if (265 <= MyWait(2) <=319) then
Event_MenuClick(Gr,6);
end; // if
end; // case
else
if (size(MyMouse(1))==0) and (size(MyMouse(2))==0) and (LastWait(1)==2) and (size(LastWait)==1) then
// {2} {{}{}}
// textout_p("New mouse event: Mouse Up After Click",Gr,1,next_line,1);
end; // if
end; // if
end; // if
```

```
if (MyWait(1)==5) then
case
if (size(MyMouse(1))==5) and (size(MyMouse(2))==5) then
if (MyMouse(1,5)==3) and (MyMouse(2,5)==7) then
//textout_p("New mouse event: Stretch",Gr,1,next_line,1);
end; // if
end; // if
end; // case
end; // if
if (MyWait(1)==7) then
case
if (size(MyMouse(1))==5) and (size(MyMouse(2))==0) then
if (MyWait(1)==7) and (MyMouse(1,5)==0) then
Event_MouseLongClick(Gr,LastMouse(1,1),LastMouse(1,2));
end; // if
end; // if
end; // case
end; // if
end; // case
end; // if
// exit PTable is Esc key
until WaitReal==K_ESC;
end;
```