`Enjoy reading the February issue of the HP-39/40g newsletter!`

In this issue we will talk about:

Letter from the editor.

Making resolutions is one thing. Keeping them is another ☺. You've probably noticed that this newsletter is again late. But when making it I didn't thought that there was going to be so much information this month ☺. This edition is packed, packed!! We have new upcoming programs, an interview with NODA (some new questions asked) and we learn everything about lists. The program of the month is super duper and the lay-out of this newsletter is changed a little bit again. But enough talk from me... Let's read the HPMN.

Sincerely, your editor, Michaël

Program of the month.
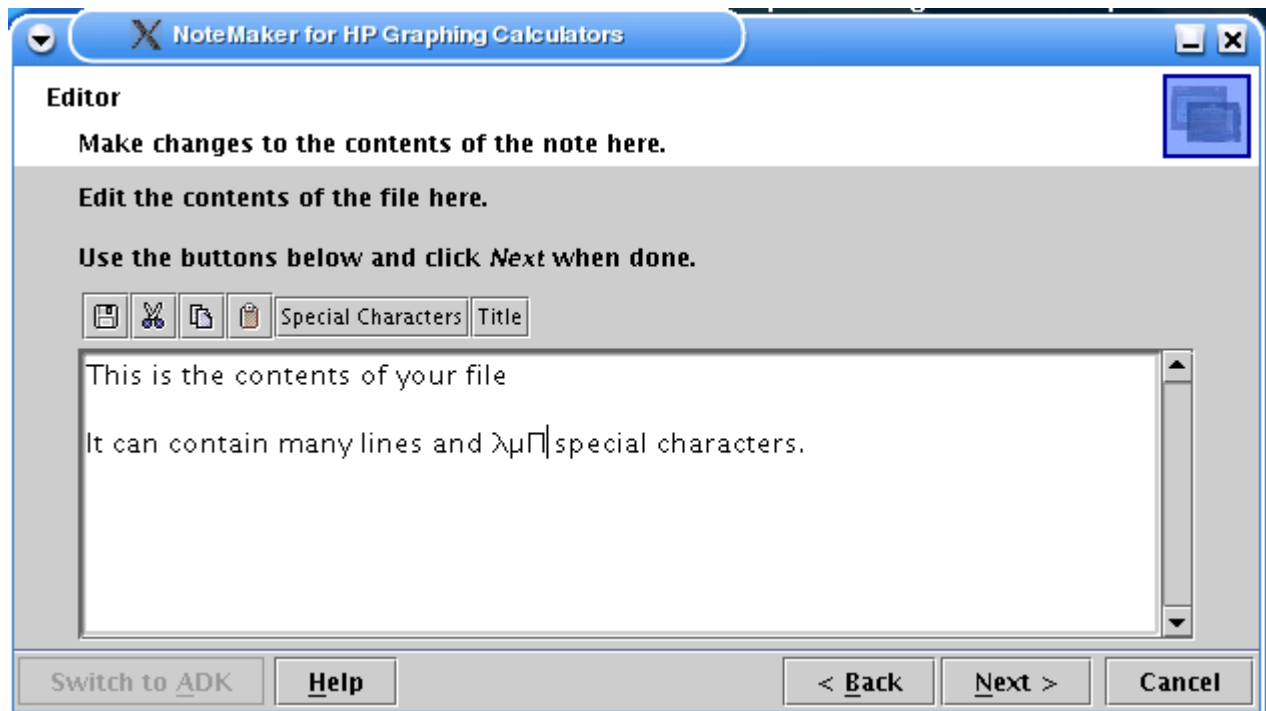
This month the winner is 'NoteMaker'! The maker itself says that it is the program where people have been asking for. And I have to admit, I didn't ask for it, but I'm sure glad that it's here ☺. I think that everybody who owns a HP-39/40g, made at least once in his life a note with it. Small notes you can do easily on your calculator. However for long notes the HP-39/40g's keyboard, like any calculator keyboard, is not ideal. With this great program you can just do it on your computer. However notice that you must have java 2.0 or higher to run it. With this said the only disadvantage is also mentioned. For the rest it does what it should do and it comes in cool package. What more do you ask for? Nothing, I thought so…

Specs:

System: HP 39(+)/40g.
Author: Joshua King
Download where: http://www.geocities.com/jking_ok/notemaker/notemaker060.zip
Site of the maker: http://www.geocities.com/jking_ok/index.html

Upcoming programs.

Well someone from the French community tries to convert the **Mig player** to the HP-39/40g. Mig is a program that enables you to play polyphonic sounds on your calculator. There was already a beta version 'released'. But unfortunately it didn't work well (you couldn't hear anything ☺). Because the person who converts it doesn't have a HP-39/40g I think you shouldn't get your hopes up too high.

On the other hand Alan Lark gave me a note that he was working on **HP Piano X**. This time it supports Sys-rpl format! And he's also trying to implement real-time recording. I think it's very cool that my favourite tool when programming gets an update. I'm so happy ☺.

The French community is looking for people who want to make a game based on a game on the internet. The game is a puzzle game with pipes (the link is: http://www.telescope-game.com, you can play the game even off-line if you want). If you are interested in making it or just want to know in what state the game is go to forum of HP-Network: http://www.hp-network.com/forum/list.php?f=1. Look for (RE:) Un jeu super cool… sur HP???. Like you could guess the website is in French.

If you don't want to make a whole game or just think that this game sucks, but you are still in an Assembly/Sys-rpl programming mood then there are other options for you. NODA has a lot of unreleased games that still have bugs in them but shouldn't be that hard to solve them. Those games are:

- **Master of the grass** (a bug when one quits)
- **Novanoid** (key of the sound that doesn't work)
- **SuperCar** (bug when you quit playing)
- **Defusing** (you have to convert the user-rpl part)
- **Hapaman** (you have to convert the Sys-RPL part)
- **Jailai** (You still have to convert the keys)
- **Jezz-Ball** (Only works in a library)

Write him an e-mail if you are interested in getting the source of one of these games.

You can also ask Julien Meyer for the sources of Wario-HP and DR-Mario to convert those to the HP-39/40g if you want.

Interview with NODA.

Interview	
Michaël	How old are you, and what of education do you have?
NODA	Hello, hello! I'm 19 years old and after my bac (note editor: ending exam in France) it was a little bit of a mess. Not taking lessons for my audiovisual BTS, I went to the fac. I stayed there for one month (because it was a withdrawal for me), and after that I went finally to Cannes to make my audiovisual BTS. I stopped three months later because of my girlfriend who was at Grenoble (and who left me a week after), after that I tried to retake the fac, but it started working a little bit on my nerves so I ended by doing a half-year nothing.
Michaël	Where do you live?
NODA	I live in France at Grenoble (Echirolles to be precise)
Michaël	What calculators do you own?
NODA	A HP-40g and... that's all! I have had a TI-81 on which I made my first text-games, but it was actually my sisters' and in the end she wanted it back.
Michaël	What was the first program you ever wrote?
NODA	Ola, that goes way back, I don't remember it anymore! I started programming in Basic around the age of 6-7 years, because my uncle had given my sister a computer (if you can call it that), a Thomson TO-8, and because she didn't understand a thing about it it was me who helped her out :o)
Michaël	What do you use your calculator for the most?
NODA	Well for the moment I almost don't use it anymore. But back when I was still on school I used it to calculate everything what the teacher wanted me to do by hand and to keep the lessons interesting.
Michaël	Why did you start programming calculators and how did you learn it?
NODA	Well I already programmed for a time in basic on our TO-8, and in qbasic when we got our first computer (an unbelievable 386sx ☺). From the moment I got in college and when my sister passed her TI-81 to me, I saw that you could program it too, so inevitably I decided to learn programming for the TI-81. I have learned everything by myself by reading books and sources, by testing programs and re-testing programs...
Michaël	If you have any advice for beginning Sys-rpl and Assembly programmers, what would it be?
NODA	What ever you do don't get discouraged. When I started programming on the HP-40g, nobody knew anything about this new calculator which just came out. I have tested a lot of instructions, one for one, to know what they did and to see how they worked in the beginning. Yes, that was hard work but with perseverance I got there. Assembly is a language that scares you in the beginning (one says that it's crypted when you read a source) and when I started converting games, I almost didn't know anything. Lilian told me briefly how I should do it, so I tested what he said by compiling all the games for the HP-48 which I could find and looked if it worked. Little by little I started to understand how it worked, without doing much effort. And today I know how to code in ASM (I secretly made a doom-like game, but as many of my projects I didn't find time to finish it). The only bad thing is that it takes time to learn. Don't start with to ambitious projects, you have to start smoothly and then everything will slip in place.
Michaël	Are there any programmes you want to see on your calculator?
NODA	The grey-scale mario and the grey-scale Bomberman from the HP-48. I've tried to convert them for a long time, but the first one is impossible without having the source and the second uses to much routines and calls in the un-supported area of the HP-48g which makes the conversion a whole lot harder.
Michaël	What do you see for the future of HP?
NODA	Well when there are enough motivated users, there is a future for HP!! And with the new HP-49g+, there are enough new programmers that are going to start programming.

Michaël's cosy programmers' corner.

Hi this month we will learn everything about lists. Let's start right away:

Make an aplet with the name 'Lists' and the same file name in WSR 1.1b. Let's begin with making two variables called List1 and List2. Scroll down till the beginning of the program and delete the usual lines. Now write the code to clear the screen.

You can basically store everything in a list! Ranging from BINTS to strings, you even can store lists inside lists. Well let's try to store `% 1.1`, `"Colin"` and `ONE` in List1. So write under `HARDBUFF ZEROZERO BINT_131d SIXTYFOUR GROB!ZERODRP:` `{ % 1.1 "Colin" ONE } List1!`. Now let us try to store a list inside a list. Write under the previous code `{ { %0 %0 } { %1 % 2.1234568 } } List2!` It's isn't that hard: open a list with '{' put spaces between different elements and quit the list with '}'. You're code should be:

```
HARDBUFF ZEROZERO BINT_131d SIXTYFOUR GROB!ZERODRP

{ % 1.1 "Colin" ONE } List1!
{ { %0 %0 } { %1 % 2.1234568 } } List2!
```

Now that we have stored something in List1 and List2 we are going to display the content of those lists on the screen. So take the values out of those lists (it's the same as taking a value out of variable). Make strings from them and display them on row 1 and row2. You're code should look like this (ignore the comment):

```
HARDBUFF ZEROZERO BINT_131d SIXTYFOUR GROB!ZERODRP

{ % 1.1 "Colin" ONE } List1!
{ { %0 %0 } { %1 % 2.1234568 } } List2!

List1@ DO>STR
DISPROW1
* Notice that the bint is displayed as UNKNOWN
List2@ DO>STR
DISPROW3
WaitForKey
```

Compile your aplet. Transfer it to your HP-39/40g and run it. Notice that the BINT in our first list is displayed as UNKNOWN.

Be able to make lists is a nice thing doing something useful with them is another. E.G.: let's say we want to have the second value of the first list (Colin). We use `NTHCOMPDROP`. `NTHCOMPDROP` is a one word command for `NTHELCOMP DROP`. `NTHELCOMP` needs two arguments namely a list and a BINT. It checks if the BINT isn't bigger than there are elements in the list and puts FALSE (if the BINT wasn't bigger) or TRUE (if the BINT was bigger) on the stack. Then if the case was FALSE it's puts the object on the stack that was on the place specified by the BINT. Since I'm not interested in getting TRUE or FALSE I use `NTHELCOMP DROP`. But because `NTHCOMPDROP` is a little bit smaller in size I use that instead. We want to display Colin on row 3 till a key is pressed. Colin is the second object of list 1. So we need `List1@` and `TWO` as arguments. So write `List1@ TWO NTHCOMPDROP`. Now display it on row 3 (remember Colin is already a string) and add a waitforkey. Your code should look like this:

```
List1@ TWO NTHCOMPDROP
DISPROW3

WaitForKey
```

So that works. What I want to do now is replace the % 1.1 in list 1 with a string hello. For that we have the command `PUTLIST`. `PUTLIST` needs three arguments an object, a BINT and a list. It replaces the object that is on the place defined by the BINT in the list you have defined with the object you have defined. So we write `"Hello" ONE List1@ PUTLIST`. Now store the result back in List1. The code should look like this, notice that I have added code to display the list on row 1.

```
DISPROW3
WaitForKey

HARDBUFF ZEROZERO BINT_131d SIXTYFOUR GROB!ZERODRP

"Hello" ONE List1@ PUTLIST List1!
List1@ DO>STR
DISPROW1
```

Let's store something totally new in List 1. E.g.: `{ %2 } List1!`. Now I want to add a different list to my List 1. So now I need to use the command `&COMP`. `&COMP` needs two arguments a list and a second list. It adds the second list to the first one. We want to add `{ %3 % 4.123 }` to our list 1. So our first argument is List 1 and our second { %3 % 4.123 }. Write `List1@ { %3 % 4.123 } &COMP`. Store the result back in List 1. Your code so far (I added again some code to display the list):

```
{ %2 } List1!

* Let's add a new list to list1 *

List1@ { %3 % 4.123 } &COMP List1!
List1@ DO>STR DISPROW3

WaitForKey
```

As a last exercise we are going to count how many elements there are in our list. We use the command `LENCOMP` for that. It uses only one argument… Guess which… Yes indeed a list. So write `List1@ LENCOMP`. The number of elements that are in the list is now put on the stack. But it isn't a real number, but a BINT. So to display it we have to make a real number from (and also make a string of it, of course). The code (again I felt free to add the code to display the number on the screen):

```
HARDBUFF ZEROZERO BINT_131d SIXTYFOUR GROB!ZERODRP

List1@ LENCOMP
UNCOERCE DO>STR DISPROW1

* If you also want the list to be returned use DUPLENCOMP
```