

НАС

G. de Montmollin, FOSDEM 2020

HAC is...

- “**HAC** Ada **C**ompiler”
- “**H**ello-world **A**da **C**ompiler”
- “**H**acker’s **A**da **C**ompiler”



HAC is NOT an Ada compiler

HAC's scope & goals

- **Experimental**
- **Incomplete** (there are enough “big” compilers).
- **Goal:** provide a simple, quick compiler
 - for beginners or non IT specialists
 - for small projects
- Work was already done in that direction.
Idea: leverage it!



HAC's development

Pimp My Compiler !

- Renovate a rusty project: **SmallAda**.
- **1st step**: get rid of Pascal (**SmallAda** was written in Pascal).
 - Biggest issue: Pascal's dialect fragmentation, affecting **SmallAda** too:
 - 1 **SmallAda** version in Mac Pascal
 - 1 **SmallAda** version in Turbo Pascal (DOS).
 - Have a portable, time-resistant source base. Ideal choice: **Ada**!
 - Nice consequence: an **Ada-in-Ada** compiler.

SmallAda's history

From **Pascal-S** to **SmallAda**



~**1970**: Pascal-S: a Pascal compiler contained a single Pascal program, written by Niklaus Wirth himself !

~**1986**: Co-Pascal

~**1990**: SmallAda

Then: nothing! Abandoned software, 2 versions, bound to DOS or Mac OS < X.

~~SmallAda~~ HAC's history

1999: First attempt to translate SmallAda from Pascal to Ada, using **P2Ada**.

Issue: the WITH statement (in Pascal).

```
type T = record x, y: Integer end;  
var r: T;
```

```
with r do begin p(x); q(y) end;    ↔    p(r.x); q(r.y);
```

2009: Made **P2Ada** smarter: understands custom types, among them, records, then WITH's.

2013: January 24th: **Day Zero** of **HAC**. Hello World, Fibonacci, sorting demos and few other tests work!



HAC's development

- **2nd step (in progress)**: make **HAC** support correct Ada. Basically, remove remaining bits of Pascal rules.
Example: Pascal features implicit type conversions.



- **3rd step (open-end)**: expand **HAC**, depending on random needs (“script-like”, small code development, teaching)...
Feedback is welcome!



HAC's features

- Featured: \pm “Pascal subset”, plus tasks:
 - custom types (partially)
 - recursion
 - nesting
- Not yet featured:
 - packages
 - generics
 - ...

HAC's characteristics

- **Build time** of the full HAC compiler & VM interpreter, by GNAT: less than 2 seconds
- **Build time** of any small example, by HAC: less than 1/100 second
- **System dependency:** none
- Ada source input: any stream (file, editor data, web stream, zip archive, ...)
- **Targets:**
 - *Currently:* p-code Virtual Machine
 - *Could be:* through abstraction: dedicated targets, LLVM?, ∞ ...

Where to find **HAC** ?

HAC is free, open-source (MIT license)

- **SourceForge:**
<https://sourceforge.net/projects/hacadacompiler/>
- **Github:** <https://github.com/zertovitch/hac>

Projects related to **HAC**

LEA: Lightweight Editor for Ada

- <https://sourceforge.net/projects/l-e-a/>
- <https://github.com/zertovitch/lea>



Pascal-to-Ada

- <https://sourceforge.net/projects/p2ada/>
- <https://github.com/zertovitch/pascal-to-ada>

```
begin
  v := "ZYXWVUTSRQPONMLKJIHGFEDCBA";
  c2 := "r1";
  max := 26;
  Put_Line ("Merge Sort");
  New_Line;
  Put_Line ("String at start:");
  Put_Line ("-----");
  for k in 1..26 loop
    Put (v(k));
  end loop;
  New_Line;
  New_Line;
  --
  cur_length := 1;
  while cur_length < max loop -- New phase
    temp_array := v;
    for k in 1..26 loop
      Put (temp_array(k));
    end loop;
    New_Line;
    left := 1;
    m := 1;
    while left <= max loop -- Find pair of subarrays
```

Console

Merge Sort

String at start:

ZYXWVUTSRQPONMLKJIHGFEDCBA

ZYXWVUTSRQPONMLKJIHGFEDCBA

YIzWlXrUUVeSlTrQlRcOlPrMlNcKlLrIlJcGlHrElFcClDrAlBr

YZWXUVSTQROPNMKLIJGHFEDCBA

WlXlYrZsSlTlUvOlPlQrRrKlLlMnNrGlHlIrJcClDlErFrArBr

WXYZSTUVO PQ RKL MN GHIJCDEFAB

SlTlUvVlWlXrYrZrKlLlMlNlOlPrQrRrClDlElFlGrHlIrJcArBr

STUVWXYZKLMNOPQR CDEFGHIJAB

WlLlMlNlOlPlQlRlSrTrUvVlWlXrYrZrAlBlCrDrErFrGrHlIrJr

KLMNOPQRSTUVWXYZAB CDEFGHIJ

AlBlClDlElFlGlHlIlJlKlLrLrMlNrOrPrQrRrSrTrUrVrWlXrYrZr

Result of Merge Sort:

ABCDEFGHIJKLMN OPQRSTU VWXYZ

