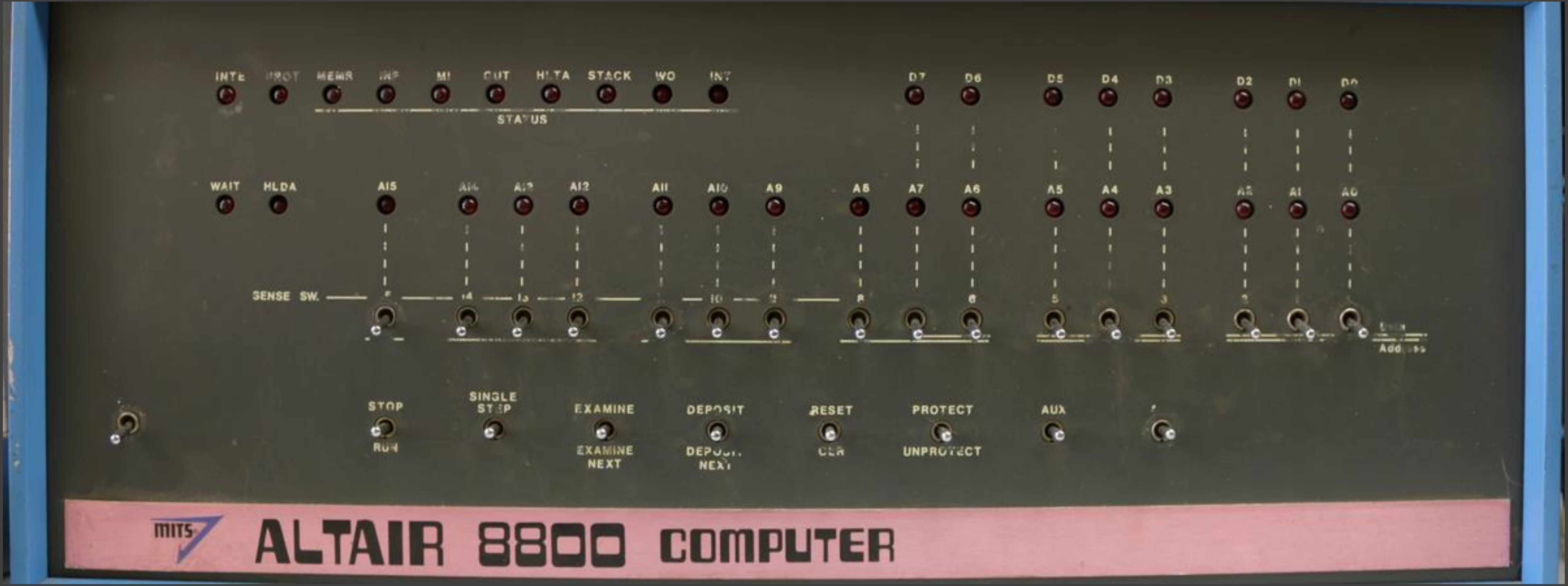
Low-code development

Dennis Vroegop

In the old days…

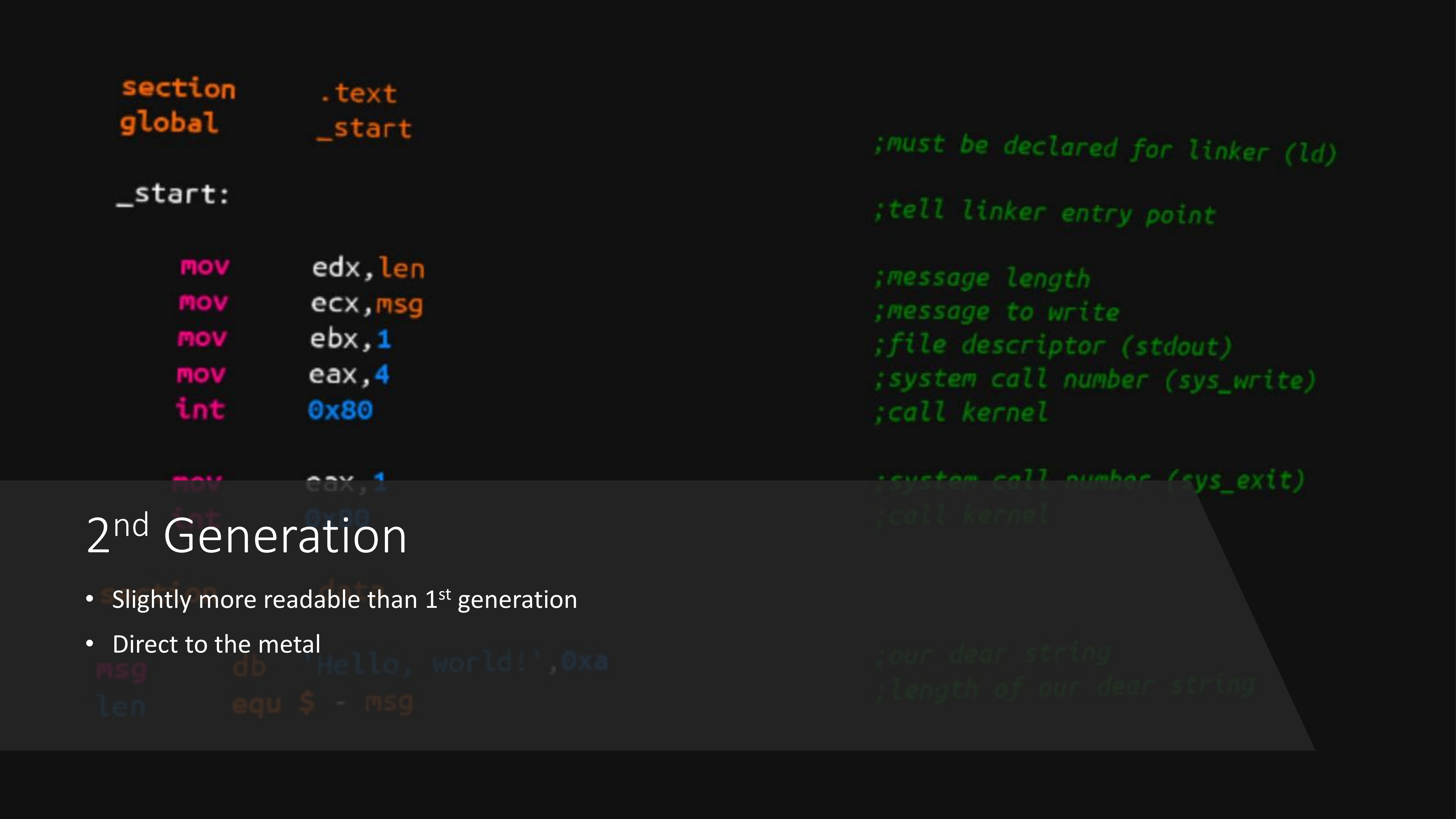
Language generations

- **1st generation:** Machine language.
- **2nd generation:** Assembly language
- **3rd generation:** High-level language (C, C++, C#, Java, JavaScript)
- **4th generation:** Very high-level language (SQL, Uniface, PowerBuilder)



1st Generation

- True binary: zero / one, on / off
- Not for the faint of heart…



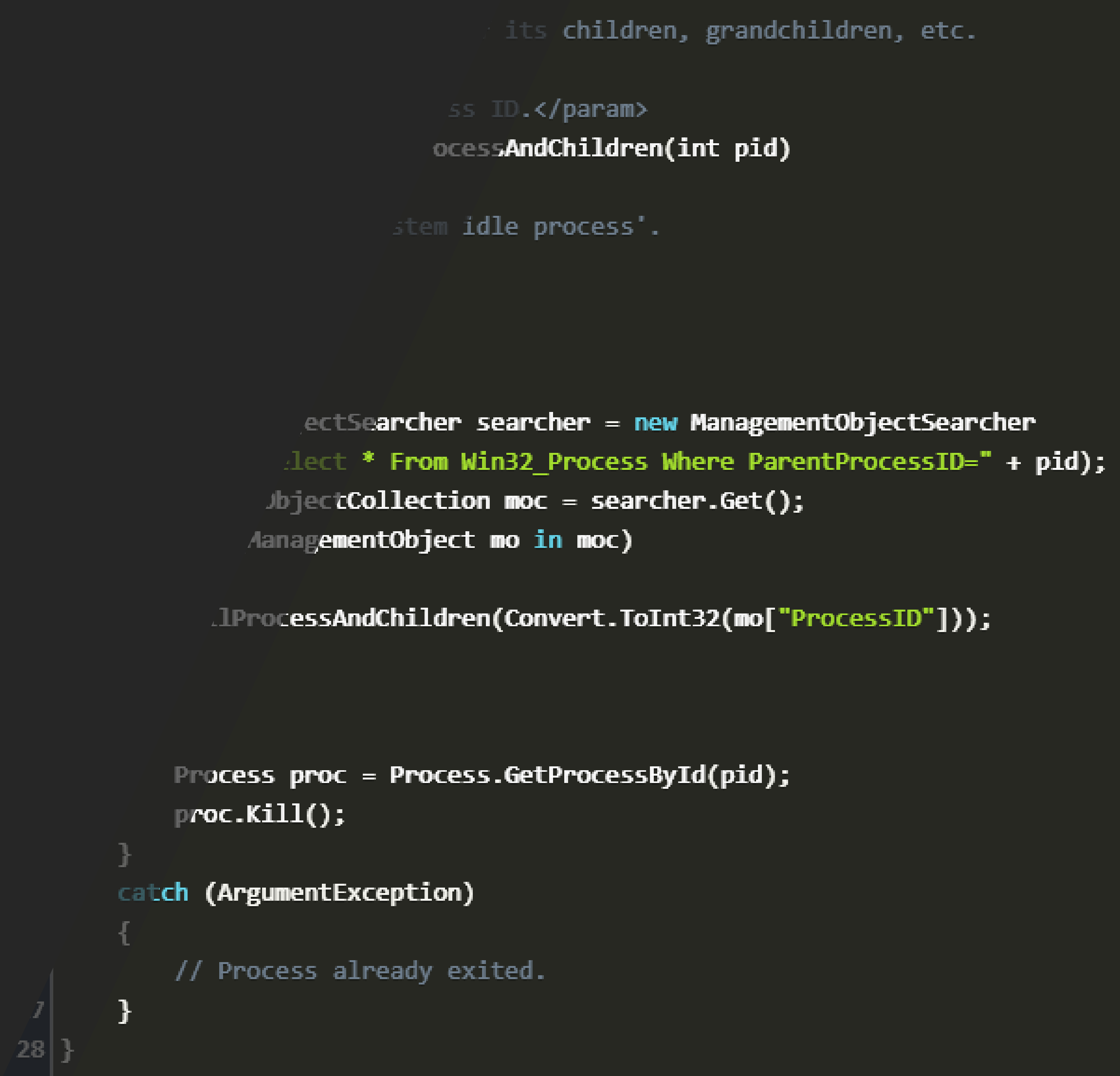
```
section     .text
global      _start                          ;must be declared for linker (ld)

_start:                                     ;tell linker entry point

    mov     edx,len                         ;message length
    mov     ecx,msg                         ;message to write
    mov     ebx,1                           ;file descriptor (stdout)
    mov     eax,4                           ;system call number (sys_write)
    int     0x80                            ;call kernel

    mov     eax,1                           ;system call number (sys_exit)
    int     0x80                            ;call kernel

section     .data

msg     db  'Hello, world!',0xa             ;our dear string
len     equ $ - msg                         ;length of our dear string
```

2nd Generation

- Slightly more readable than 1st generation
- Direct to the metal

3rd Generation

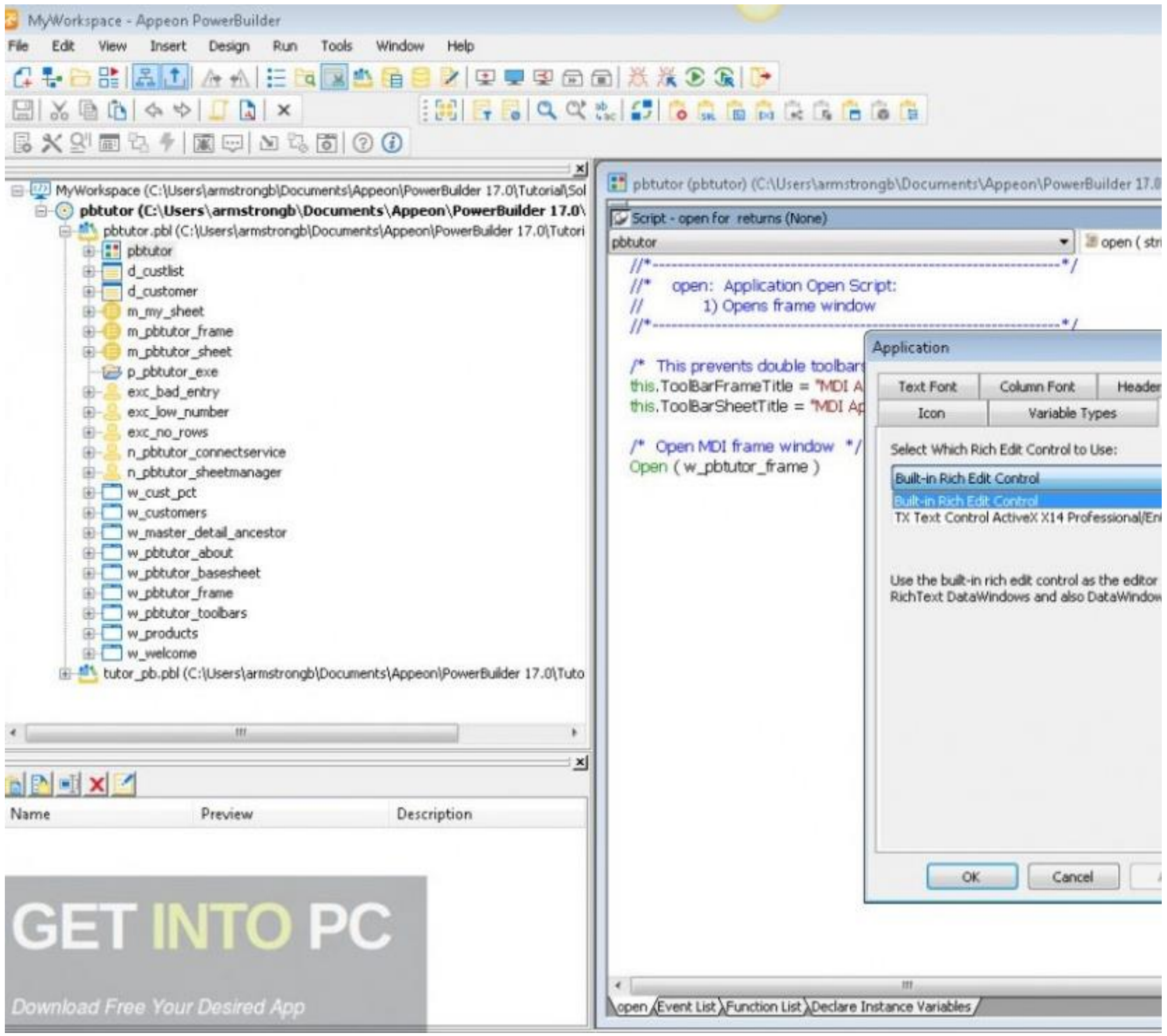
- Most used today
- Very flexible
- Requires coding knowledge



```
                         its children, grandchildren, etc.

                        ss ID.</param>
                       ocessAndChildren(int pid)

                     stem idle process'.




                 ectSearcher searcher = new ManagementObjectSearcher
                lect * From Win32_Process Where ParentProcessID=" + pid);
               ObjectCollection moc = searcher.Get();
              ManagementObject mo in moc)

            lProcessAndChildren(Convert.ToInt32(mo["ProcessID"]));




        Process proc = Process.GetProcessById(pid);
        proc.Kill();
    }
    catch (ArgumentException)
    {
        // Process already exited.
    }
}
```

4th Generation

- Business oriented
- Tight vendor lock-in
- Limited support
- Limited knowledge
- Coding in proprietary tools


MyWorkspace - Appeon PowerBuilder
File Edit View Insert Design Run Tools Window Help
Name Preview Description
open Event List Function List Declare Instance Variables
Application
Text Font Column Font Header
Icon Variable Types
OK Cancel

```
MyWorkspace (C:\Users\armstrongb\Documents\Appeon\PowerBuilder 17.0\Tutorial\Sol
  pbtutor (C:\Users\armstrongb\Documents\Appeon\PowerBuilder 17.0\
    pbtutor.pbl (C:\Users\armstrongb\Documents\Appeon\PowerBuilder 17.0\Tutori
      pbtutor
      d_custlist
      d_customer
      m_my_sheet
      m_pbtutor_frame
      m_pbtutor_sheet
      p_pbtutor_exe
      exc_bad_entry
      exc_low_number
      exc_no_rows
      n_pbtutor_connectservice
      n_pbtutor_sheetmanager
      w_cust_pct
      w_customers
      w_master_detail_ancestor
      w_pbtutor_about
      w_pbtutor_basesheet
      w_pbtutor_frame
      w_pbtutor_toolbars
      w_products
      w_welcome
    tutor_pb.pbl (C:\Users\armstrongb\Documents\Appeon\PowerBuilder 17.0\Tuto
```

pbtutor (pbtutor) (C:\Users\armstrongb\Documents\Appeon\PowerBuilder 17.0

Script - open for returns (None)

pbtutor ▾ open (str

```
//*-----------------------------------------------------------------*/
//*   open:  Application Open Script:
//          1) Opens frame window
//*-----------------------------------------------------------------*/

/*  This prevents double toolbar
this.ToolBarFrameTitle = "MDI A
this.ToolBarSheetTitle = "MDI Ap

/*  Open MDI frame window  */
Open ( w_pbtutor_frame )
```

Select Which Rich Edit Control to Use:

Built-in Rich Edit Control

Built-in Rich Edit Control
TX Text Control ActiveX X14 Professional/En

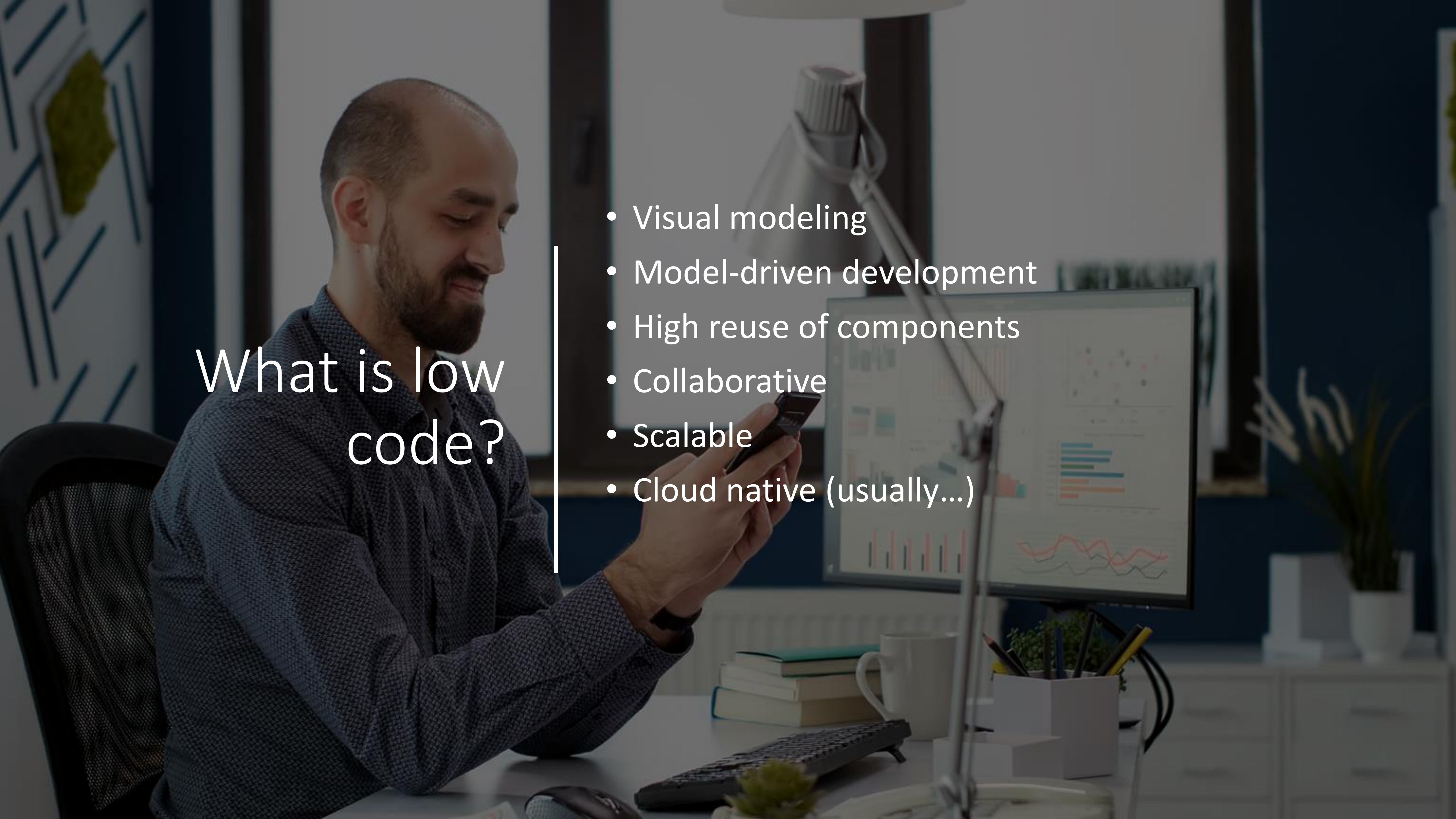
Use the built-in rich edit control as the editor
RichText DataWindows and also DataWindow

Trouble on the horizon

- Tools get better,
- … but demands are growing
- … and good people are scarce
- … and we need to be everywhere
- … time-to-market needs to be shorter

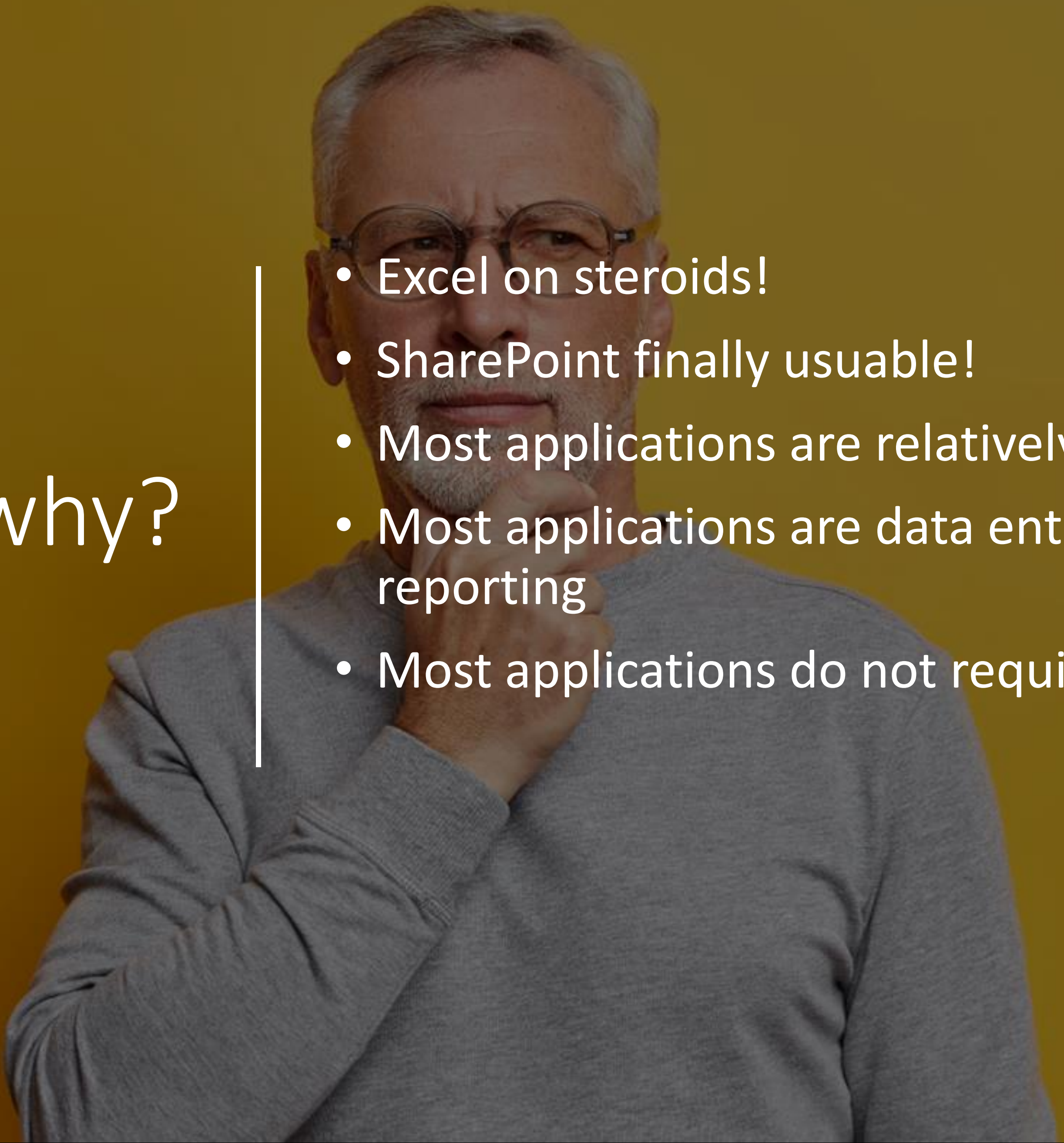
Low-code: a solution?

- Is there light at the end of the tunnel?



What is low code?

- Visual modeling
- Model-driven development
- High reuse of components
- Collaborative
- Scalable
- Cloud native (usually…)



But why?

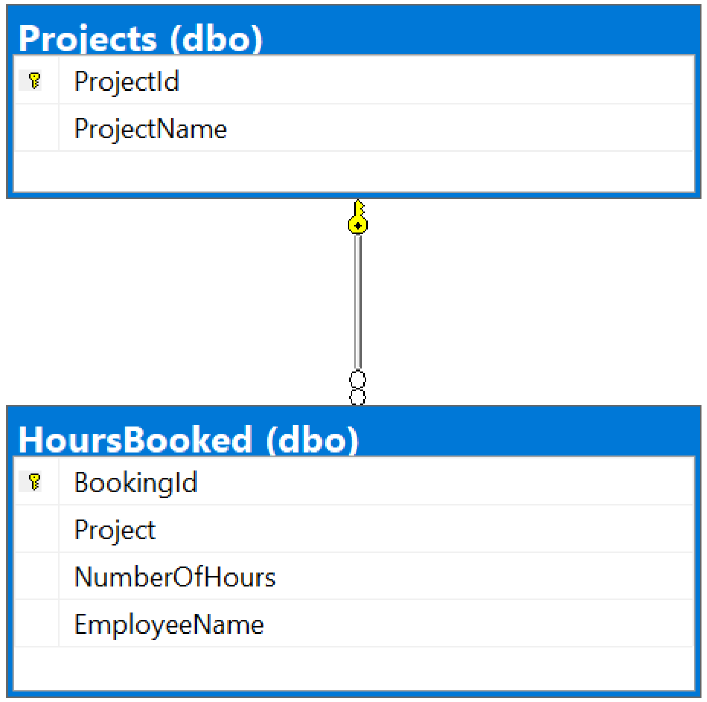
- Excel on steroids!
- SharePoint finally usuable!
- Most applications are relatively straightforward
- Most applications are data entry, validation, and reporting
- Most applications do not require IT Wizards

The demo case

- Hour management
- Data is stored in enterprise db
- Needs custom-code validations
- Needs to be finished today!

The database *(if you can call it that…)*

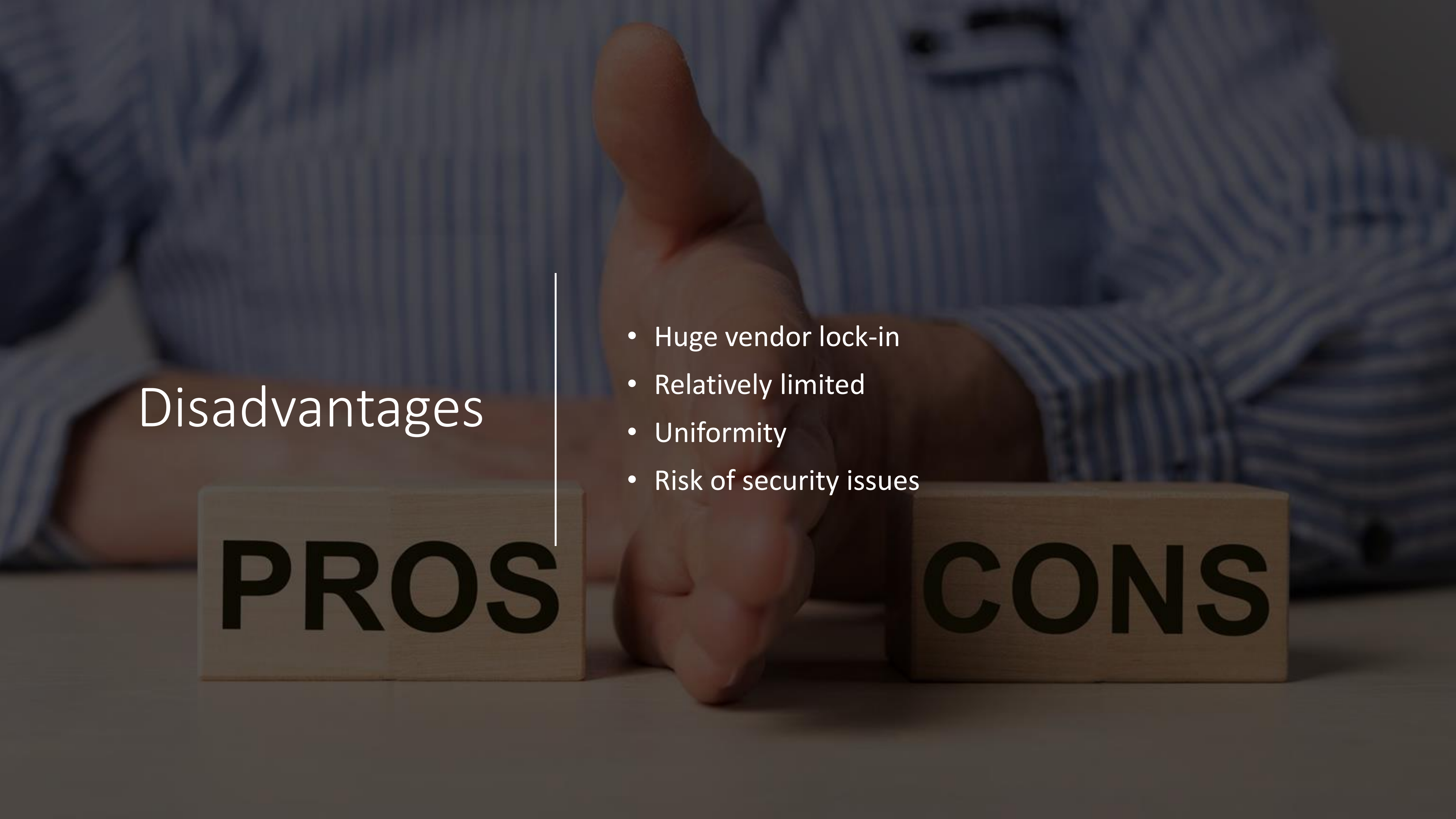
- Projects: all projects we have
- HoursBooked: all employees and the total number of hours worked per project



Show me the goodies!

Advantages

- Learning curve
- Leverage Micro Services
- Component reuse
- Use knowledge where needed
- Quick deployments
- Automatic updates from vendors

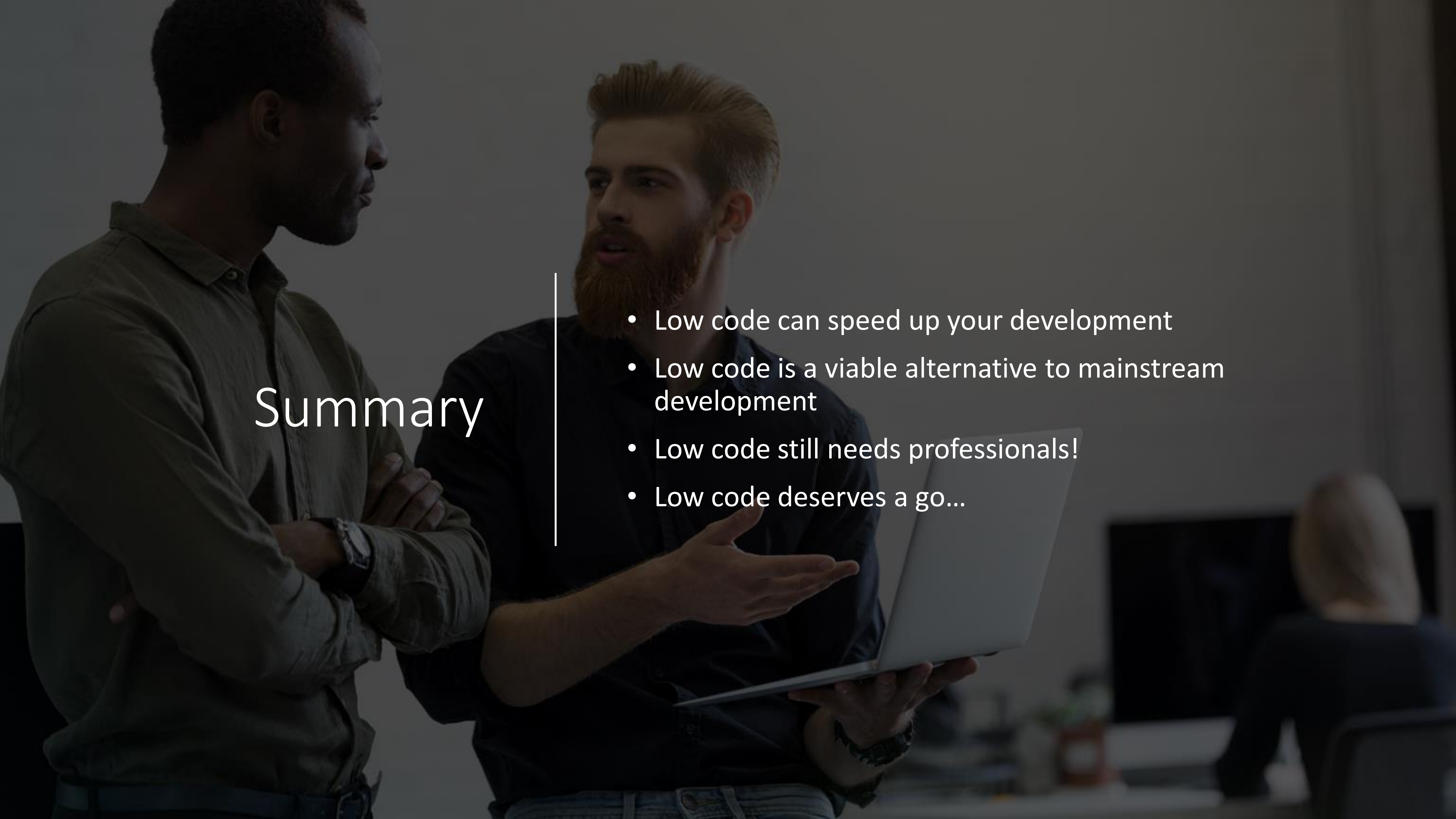


Disadvantages

- Huge vendor lock-in
- Relatively limited
- Uniformity
- Risk of security issues

Case study: B&S by Codeless

- Full ERP system
- Controlling robots in the warehouse
- EDI coupling with suppliers
- Interface 3rd party financial software
- Fully cloud enabled
- Used on almost all continents
- Secure and audited



Summary

- Low code can speed up your development
- Low code is a viable alternative to mainstream development
- Low code still needs professionals!
- Low code deserves a go…

Q&A

For questions:
dennis@destrato.com
https://www.dennisvroegop.com