Coding in 2021

Evolution of Conventions

Coding Conventions

What?

- A set of guidelines for a specific programming language that recommend programming style, practices, and methods for each aspect of a program written in that language

Why?

- Readability, understandability, ease of maintenance

How?

- Enforced socially, or with tools (style cop)

Where did conventions come from?

Well, I have a theory… back in the mists of time,

Software was written in a very predefined manner!

Some examples:

Assembly

```
08048918     pushl  %ebp
08048919     movl   %esp,%ebp
0804891b     subl   $0x4,%esp
0804891e     movl   $0x0,0xfffffffc(%ebp)
08048925     cmpl   $0x63,0xfffffffc(%ebp)
08048929     jle    08048930
0804892b     jmp    08048948
0804892d     nop
0804892e     nop
0804892f     nop
08048930     movl   0xfffffffc(%ebp),%eax
08048933     pushl  %eax
08048934     pushl  $0x8049418
08048939     call   080487c0 <printf>
0804893e     addl   $0x8,%esp
08048941     incl   0xfffffffc(%ebp)
08048944     jmp    08048925
08048946     nop
08048947     nop
08048948     xorl   %eax,%eax
0804894a     jmp    0804894c
0804894c     leave
0804894d     ret
```

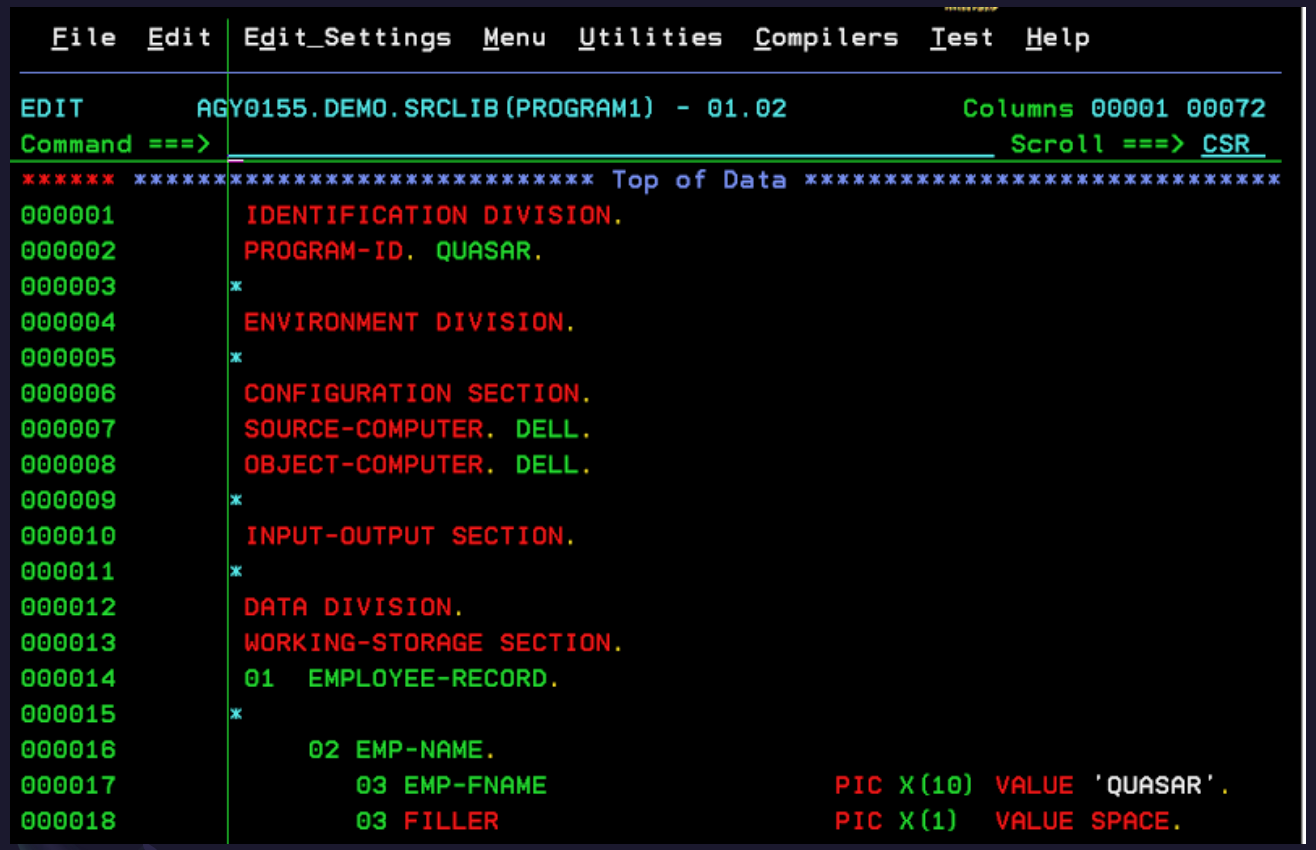
What does this do?

No idea.

Key take away is – look how strictly formatted it is!

Address column, instruction column, operand column

COBOL



COBOL has very strict rules on how you format your code.

There are SECTIONS that have to be put in order to set up the program.

As you can see, defining the employee record structure is very columnar.

RPG

```
CSR   ZASNMS     BEGSR
  *= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
  * SEND MESSAGE   TO PROGRAM Q
  *= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
  * IF FIRST MESSAGE IN CYCLE, SEND
C    N99              CALL 'YYSNMSC'
C                     PARM                ##PGVN   10      Message  q.
C                     PARM '*SAME'        ##PGRL    5      REL queue
C                     PARM                MSGID     7      Message  id.
C                     PARM                MSGF     10      Message  file
C                     PARM                MSGDTA  132      Message  data
C                     PARM '*INFO'        MSGTYP    7      Message  type
  *
  * Clear all fields for default mechanism next time.
C                     MOVEL*BLANK         MSGID            Message  Id.
C                     MOVEL*BLANK         MSGF             Message  file.
C                     MOVEL*BLANK         MSGDTA           Message  data.
```

RPG = Report Program Generator

Used on IBM Mainframes

My late Father was an RPG dev

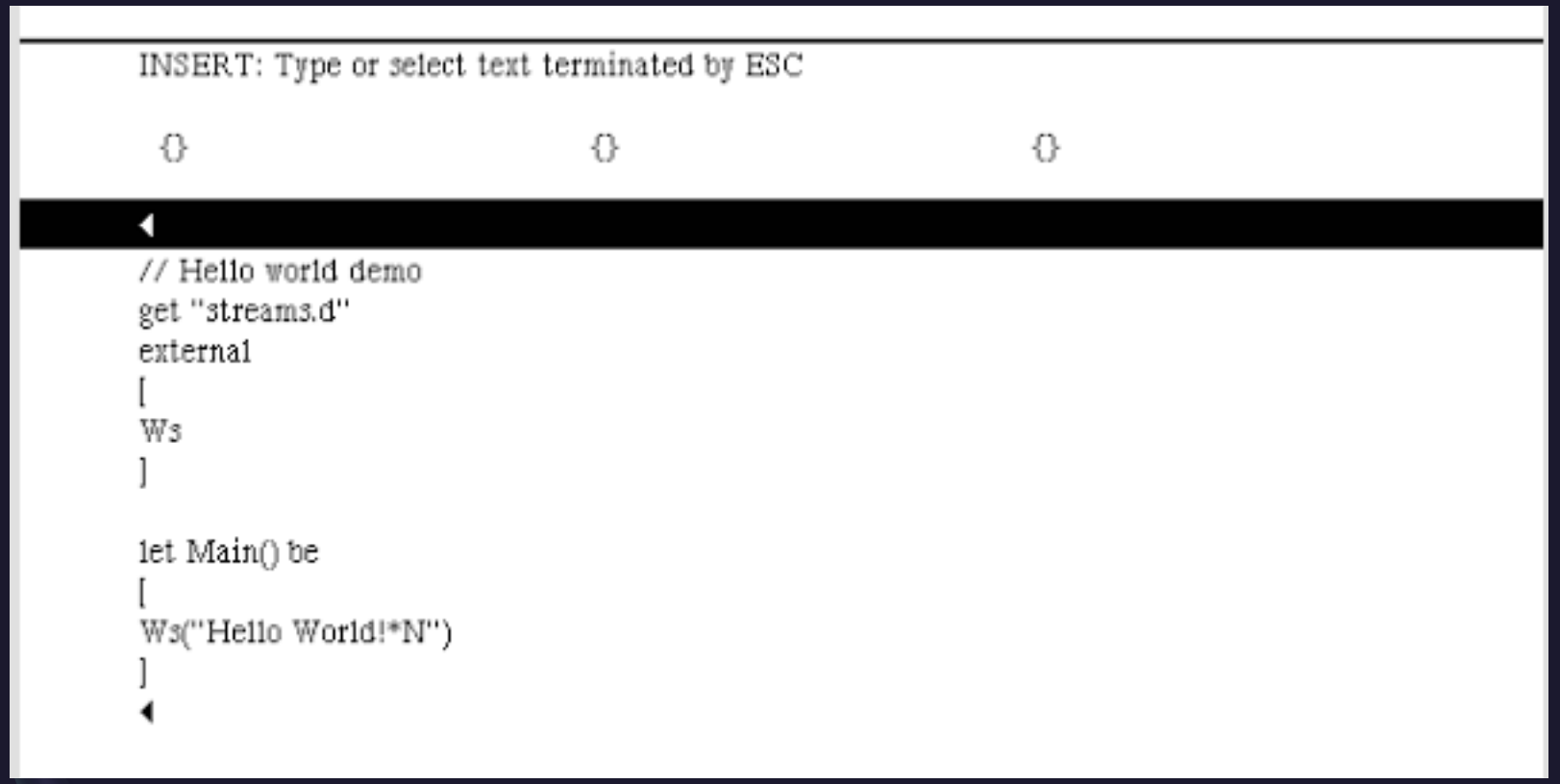
Columnar format, strict format to follow

Then high level programming languages happened

These were and are freeform!

Some examples:

BCPL (C forerunner)



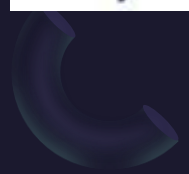
So now we're starting to look more familiar, right?

Notice how you have to have a Main (this is still a thing) but is in general a lot less structured than previous examples.

C

```c
#include "myMult.h"

void myMult(const double a[12], const double b[20], double c[15])
{
  int i0;
  int i1;
  int i2;
  for (i0 = 0; i0 < 3; i0++) {
    for (i1 = 0; i1 < 5; i1++) {
      c[i0 + 3 * i1] = 0.0;
      for (i2 = 0; i2 < 4; i2++) {
        c[i0 + 3 * i1] += a[i0 + 3 * i2] * b[i2 + (i1 << 2)];
      }
    }
  }
}
```



C code looks a lot like the C# we all know (and love right), except more bonkers.

Why are the params consts?

“…declaring function parameters const indicates that the function promises not to change these values…”

C parameters are passed by VALUE, so if you change it in the function the change will be lost so its CONVENTION to have it as const to remind the developer of this fact.

Java

```
package rentalStore;
import java.util.Enumeration;
import java.util.Vector;

class Customer {
    private String _name;
    private Vector<Rental> _rentals = new Vector<Rental>();

    public Customer(String name) {
        _name = name;
    }
    public String getMovie(Movie movie) {
        Rental rental = new Rental(new Movie("", Movie.NEW_RELEASE), 10);
        Movie m = rental._movie;
        return movie.getTitle();
    }
    public void addRental(Rental arg) {
        _rentals.addElement(arg);
    }
    public String getName() {
        return _name;
    }
```

This is Java, which to me looks like a stupid version of C#

Here we see the K&R brace convention (which for C# is bad and wrong), along with conventions of:

- A type having an upper case starting letter
- Fields have the underscore (nice)
- Method names having camel case names
- No spaces between functions

More flexible syntax and layout, but more conventions to help it be readable for Java devs

C#

```
class Class1
{
    public virtual string TestFunction()
    {
        return "Hello";
    }
}
class Class2 : Class1
{
    public override string TestFunction()
    {
        return "Bye Bye";
    }
}
class Program
{
    static void Main(string[] args)
    {
        Class2 obj = new Class2();
        Console.WriteLine(obj.TestFunction());
        Console.ReadLine();
    }
}
```

That's better ☺

Looks like a more modern Java to me. A few conventions stand out:

- Class/type names start wit ha capital letter
- Allman style braces
- Easier to read (IMHO)

This ain't the best example…

- var vs type name – discuss!
- No spaces between class definitions :-o (off to the tower with you)

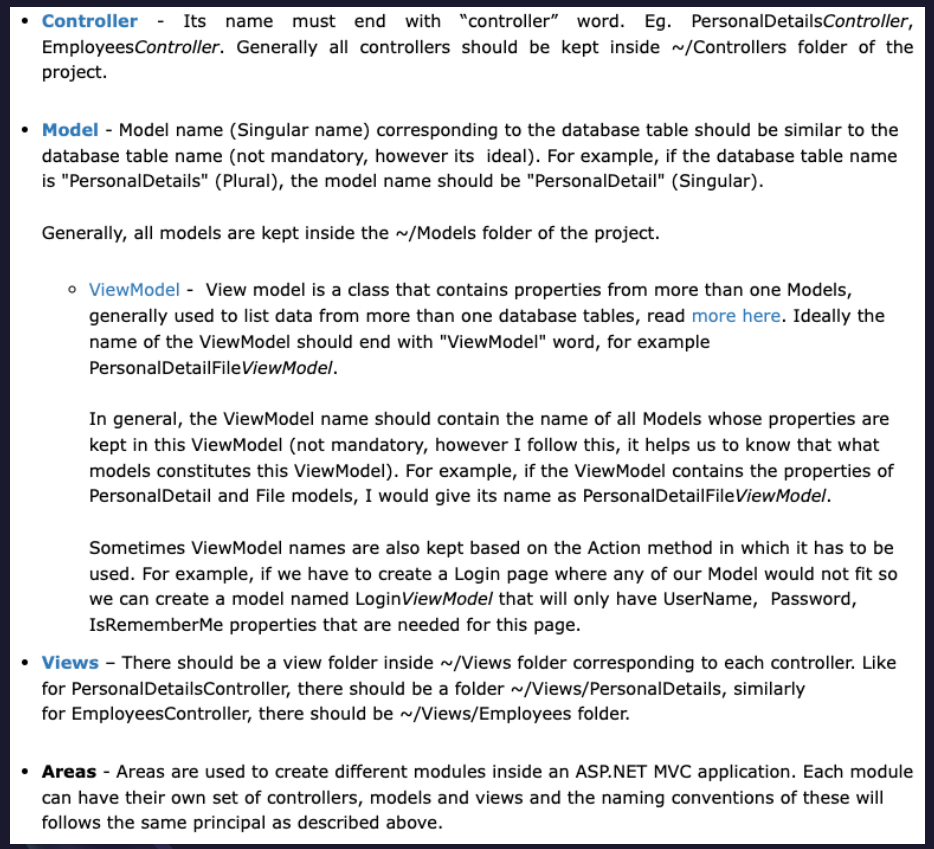
But looks a lot like the code we use in work, right?

Not just languages…

Frameworks also have conventions

Simple Example – ASP.NET MVC

ASP.NET MVC



- **Controller** - Its name must end with "controller" word. Eg. PersonalDetails*Controller*, Employees*Controller*. Generally all controllers should be kept inside ~/Controllers folder of the project.

- **Model** - Model name (Singular name) corresponding to the database table should be similar to the database table name (not mandatory, however its ideal). For example, if the database table name is "PersonalDetails" (Plural), the model name should be "PersonalDetail" (Singular).

  Generally, all models are kept inside the ~/Models folder of the project.

  - ViewModel - View model is a class that contains properties from more than one Models, generally used to list data from more than one database tables, read more here. Ideally the name of the ViewModel should end with "ViewModel" word, for example PersonalDetailFile*ViewModel*.

    In general, the ViewModel name should contain the name of all Models whose properties are kept in this ViewModel (not mandatory, however I follow this, it helps us to know that what models constitutes this ViewModel). For example, if the ViewModel contains the properties of PersonalDetail and File models, I would give its name as PersonalDetailFile*ViewModel*.

    Sometimes ViewModel names are also kept based on the Action method in which it has to be used. For example, if we have to create a Login page where any of our Model would not fit so we can create a model named Login*ViewModel* that will only have UserName, Password, IsRememberMe properties that are needed for this page.

- **Views** – There should be a view folder inside ~/Views folder corresponding to each controller. Like for PersonalDetailsController, there should be a folder ~/Views/PersonalDetails, similarly for EmployeesController, there should be ~/Views/Employees folder.

- **Areas** - Areas are used to create different modules inside an ASP.NET MVC application. Each module can have their own set of controllers, models and views and the naming conventions of these will follows the same principal as described above.

MVC conventions as per the image here.

But why?

Otherwise it gets confusing!

That’s the point isn’t it?

Conventions are SIGNPOSTS

That’s the point isn’t it?

Coding 10, 20, 30 years ago was arguably SIMPLER

(in some ways, certainly less moving parts, but each moving part was more complex than today)

There were FEWER code files per project

There were FEWER devs in the team

Agile wasn’t always a thing, so slower paced

Convention Standards

What's the best thing about standards?

There are so many to choose from!

The TEAM should agree organically – different teams may have different conventions. When you come to a new team don't fight them, maybe suggest improvements but you are the newcomer.

So to conclude...

Conventions replaced strict structure

Conventions are sign posts to communicate intent

Teams should organically agree on their team conventions