

Formation Cobol

Découverte du langage

Le COBOL (Common Business-Oriented Language) est un langage structuré principalement utilisé dans les applications commerciales et administratives. Je vais vous présenter les commandes et concepts de base pour débiter, avec des explications simples. Notez que le COBOL est très verbeux et suit une structure rigide avec des divisions spécifiques.

Vous retrouverez d'autres extrait de notre formation Cobol en libre accès sur notre site <https://cobol.pacta.com/>

Voici un guide pour débutant, qui pourra faire des essais sur les compilateurs gratuits disponibles sur le Web. Ces exemples sont envoyés par des participants sur nos forums, en cas d'erreur, envoyez-nous un mail pour apporter des rectifications.

Pacta Cobol Network[À propos](#)[Fonctionnalités](#)[Rejoindre](#)[Contact](#)[EN](#)

Réinventons COBOL ensemble

Une communauté ouverte pour apprendre, transmettre, migrer, moderniser.

[Rejoindre la communauté](#)

Pacta Cobol Network

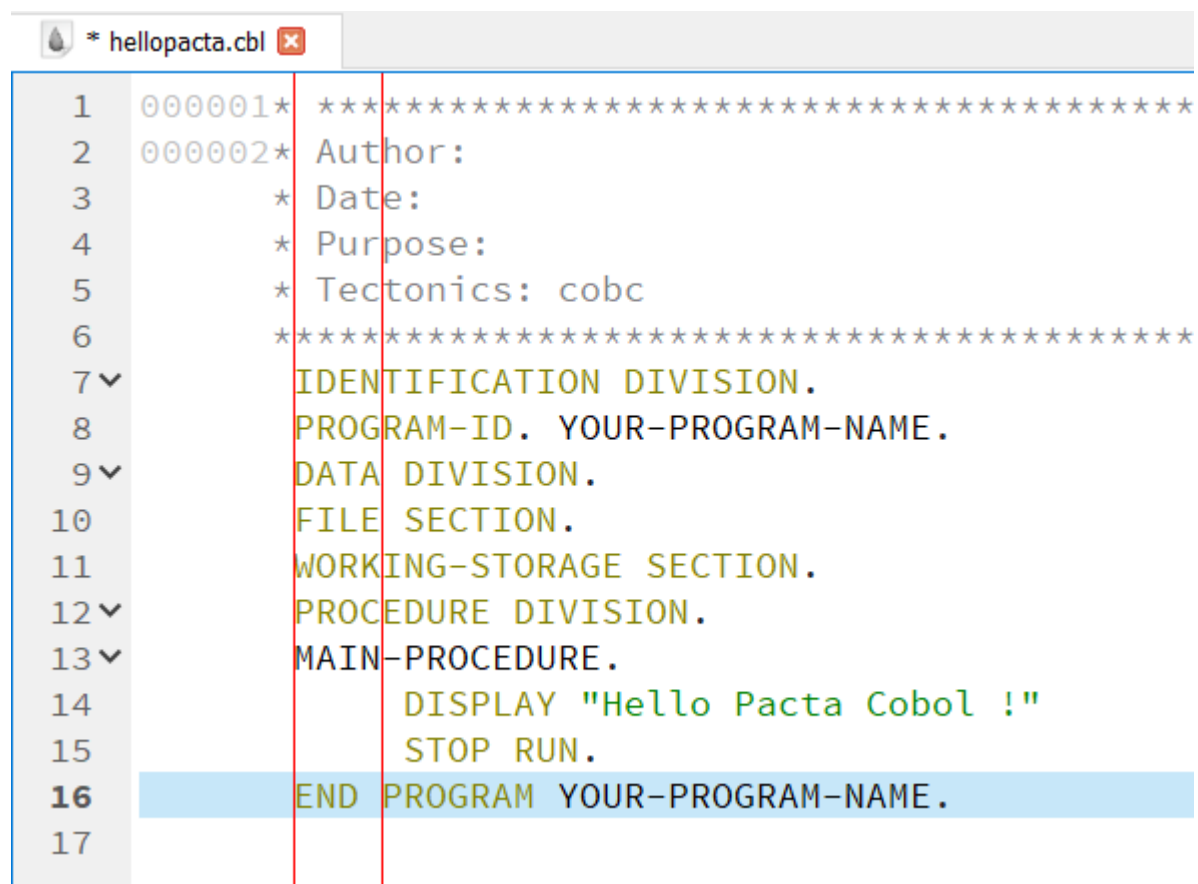
Pacta Cobol Network est un projet collaboratif pour remettre COBOL au cœur des systèmes modernes. Nous voulons documenter, transmettre et innover autour de ce langage incontournable du monde professionnel.

[GA English](#) [FR Français](#) [ES Español](#) [DE Deutsch](#) [IT Italiano](#) [PT Português](#)

Structure de base d'un programme COBOL

Un programme COBOL est organisé en **divisions**, qui sont obligatoires et doivent apparaître dans cet ordre :

1. **IDENTIFICATION DIVISION** : Identifie le programme.
2. **ENVIRONMENT DIVISION** : Décrit l'environnement (facultatif pour un débutant).
3. **DATA DIVISION** : Définit les variables.
4. **PROCEDURE DIVISION** : Contient la logique du programme.

A screenshot of a text editor window titled '* helloworld.cbl'. The editor displays a COBOL program with line numbers 1 through 17 on the left. The program text is as follows:

```
1 000001* ****
2 000002* Author:
3      * Date:
4      * Purpose:
5      * Tectonics: cobc
6      ****
7  IDENTIFICATION DIVISION.
8  PROGRAM-ID. YOUR-PROGRAM-NAME.
9  DATA DIVISION.
10 FILE SECTION.
11 WORKING-STORAGE SECTION.
12 PROCEDURE DIVISION.
13 MAIN-PROCEDURE.
14     DISPLAY "Hello Pacta Cobol !"
15     STOP RUN.
16 END PROGRAM YOUR-PROGRAM-NAME.
17
```

Exemple simple avec commandes de base

Voici un programme COBOL minimaliste qui affiche "Bonjour, monde !" :

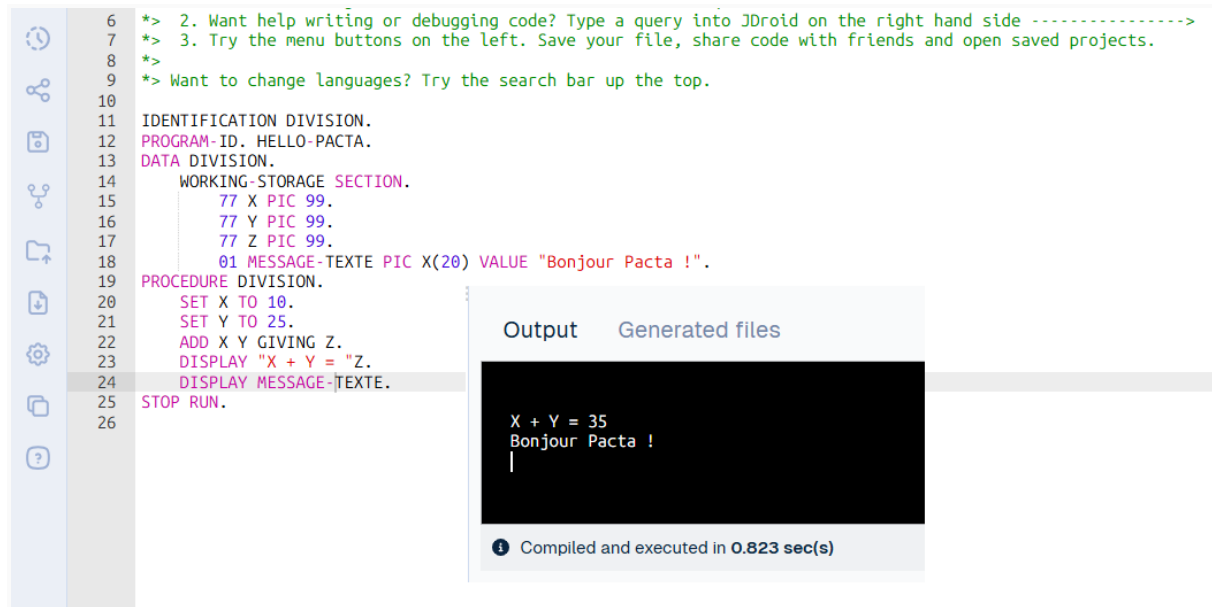
```
IDENTIFICATION DIVISION.
PROGRAM-ID. BonjourMonde.

DATA DIVISION.
WORKING-STORAGE SECTION.
01 MESSAGE-TEXTE      PIC X(20) VALUE "Bonjour, monde !".

PROCEDURE DIVISION.
```

```
DISPLAY MESSAGE-TEXTE.  
STOP RUN.
```

Voici un exemple sur un éditeur en ligne .



The screenshot shows an online code editor with a sidebar on the left containing icons for various functions like running, saving, and searching. The main area displays a program with line numbers 6 to 26. The program includes comments, an identification division, a data division with a working-storage section, and a procedure division with arithmetic and display instructions. To the right of the code, there is a panel with 'Output' and 'Generated files' tabs. The 'Output' tab is active, showing the results of the program execution: 'X + Y = 35' and 'Bonjour Pacta !'. Below the output, a status bar indicates 'Compiled and executed in 0.823 sec(s)'.

```
6  *> 2. Want help writing or debugging code? Type a query into JDroid on the right hand side ----->  
7  *> 3. Try the menu buttons on the left. Save your file, share code with friends and open saved projects.  
8  *>  
9  *> Want to change languages? Try the search bar up the top.  
10  
11 IDENTIFICATION DIVISION.  
12 PROGRAM-ID. HELLO-PACTA.  
13 DATA DIVISION.  
14     WORKING-STORAGE SECTION.  
15         77 X PIC 99.  
16         77 Y PIC 99.  
17         77 Z PIC 99.  
18         01 MESSAGE-TEXTE PIC X(20) VALUE "Bonjour Pacta !".  
19 PROCEDURE DIVISION.  
20     SET X TO 10.  
21     SET Y TO 25.  
22     ADD X Y GIVING Z.  
23     DISPLAY "X + Y = "Z.  
24     DISPLAY MESSAGE-TEXTE.  
25 STOP RUN.  
26
```

Output

```
X + Y = 35  
Bonjour Pacta !  
|
```

Generated files

Compiled and executed in 0.823 sec(s)

Explications des commandes de base

1. IDENTIFICATION DIVISION :

- `PROGRAM-ID.` : Nom du programme (obligatoire). Exemple : `PROGRAM-ID. BonjourMonde.`

2. DATA DIVISION :

- Utilisée pour déclarer des variables.
- `WORKING-STORAGE SECTION.` : Section pour définir les variables temporaires.
- `01` : Niveau de définition d'une variable (01 est le plus haut niveau pour une variable individuelle).
- `PIC X(20)` : Type de données (ici, une chaîne de caractères de 20 caractères max).
- `VALUE` : Initialise la variable avec une valeur.

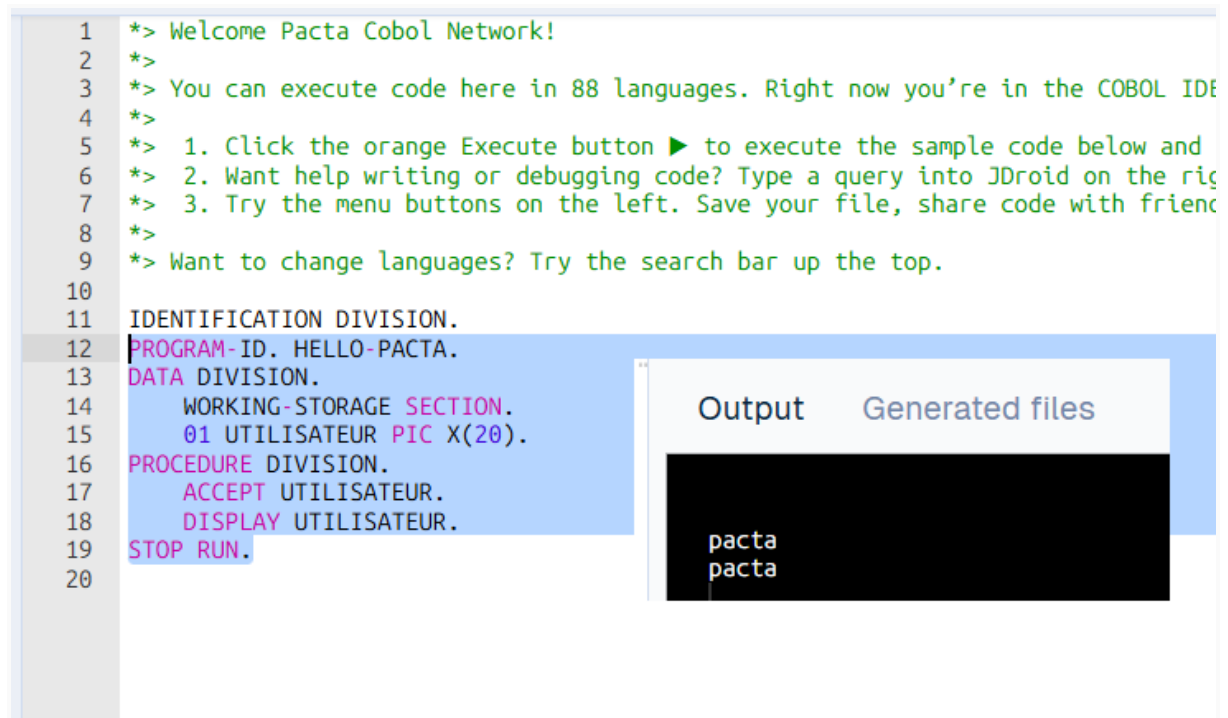
3. PROCEDURE DIVISION :

- Contient les instructions exécutables.
- `DISPLAY` : Affiche une variable ou un texte à l'écran. Exemple : `DISPLAY "Hello".`
- `STOP RUN` : Termine le programme (obligatoire pour arrêter l'exécution).

Autres commandes essentielles pour débutants

- **ACCEPT** : Lit une entrée de l'utilisateur.

```
01 ENTREE-UTILISATEUR PIC X(10).  
ACCEPT ENTREE-UTILISATEUR.  
DISPLAY "Vous avez entré : " ENTREE-UTILISATEUR.
```



The screenshot shows a COBOL IDE interface. On the left, a code editor displays the following COBOL program:

```
1  *> Welcome Pacta Cobol Network!  
2  *>  
3  *> You can execute code here in 88 languages. Right now you're in the COBOL IDE  
4  *>  
5  *> 1. Click the orange Execute button ► to execute the sample code below and  
6  *> 2. Want help writing or debugging code? Type a query into JDroid on the right  
7  *> 3. Try the menu buttons on the left. Save your file, share code with friends  
8  *>  
9  *> Want to change languages? Try the search bar up the top.  
10  
11 IDENTIFICATION DIVISION.  
12 PROGRAM-ID. HELLO-PACTA.  
13 DATA DIVISION.  
14     WORKING-STORAGE SECTION.  
15     01 UTILISATEUR PIC X(20).  
16 PROCEDURE DIVISION.  
17     ACCEPT UTILISATEUR.  
18     DISPLAY UTILISATEUR.  
19 STOP RUN.  
20
```

On the right, there is a panel with two tabs: "Output" and "Generated files". The "Output" tab is active, showing the following text:

```
pacta  
pacta
```

- **MOVE** : Affecte une valeur à une variable.

```
01 MA-VARIABLE PIC X(10).  
MOVE "Test" TO MA-VARIABLE.  
DISPLAY MA-VARIABLE.
```

```

7  IDENTIFICATION DIVISION.
8  PROGRAM-ID. HELLOPACTA.
9  DATA DIVISION.
10 WORKING-STORAGE SECTION.
11 01 VARIABLE PIC X(15).
12 PROCEDURE DIVISION.
13 MOVE "Pacta Cobol" to VARIABLE
14 DISPLAY VARIABLE.
15 STOP RUN.
16

```

Logs



Compiler



Issues



Output

C:\Users\alain\bin\helloworld.exe

Pacta Cobol

Process finished with exit code 0

- **IF** : Condition simple.

```

01 NOMBRE PIC 9(2) VALUE 10.
IF NOMBRE > 5
    DISPLAY "Nombre est supérieur à 5"
ELSE
    DISPLAY "Nombre est inférieur ou égal à 5"
END-IF.

```

```
6 *****
7 IDENTIFICATION DIVISION.
8 PROGRAM-ID. HELLOPACTA.
9 DATA DIVISION.
10 WORKING-STORAGE SECTION.
11 01 NOMBRE PIC 9(2) VALUE 10.
12 PROCEDURE DIVISION.
13 IF NOMBRE > 5
14     DISPLAY "Le nombre est superieur a 5 "
15 ELSE
16     DISPLAY "Le nombre est inferieur a 5 "
17 END-IF.
18 STOP RUN.
19
```

logs

Compiler Issues Output

C:\Users\alain\bin\hellopacta.exe
Le nombre est superieur a 5
Process finished with exit code 0

- **PERFORM** : Boucle simple.

```
PERFORM 3 TIMES  
    DISPLAY "Répétition"  
END-PERFORM.
```

Types de données de base (PIC)

- `PIC X(n)` : Chaîne de caractères de longueur `n`.
- `PIC 9(n)` : Nombre entier de `n` chiffres.
- `PIC 9(n)V9(m)` : Nombre décimal avec `n` chiffres avant la virgule et `m` après.

```
1 IDENTIFICATION DIVISION.  
2 PROGRAM-ID. HELLO-WORLD.  
3 DATA DIVISION.  
4 WORKING-STORAGE SECTION.  
5 01 TEXTE PIC X(15) VALUE "Pacta Cobol".  
6 PROCEDURE DIVISION.  
7 DISPLAY TEXTE.  
8 STOP RUN.
```

```
<command-line>: warning: "_FORTIFY_SOURCE" redefined  
<command-line>: note: this is the location of the previous definition  
Pacta Cobol
```

Conseils pour débiter

- Le COBOL est sensible à l'indentation et aux points (.) à la fin des instructions ou sections.
- Utilisez un éditeur comme OpenCOBOL (maintenant GnuCOBOL) pour tester tes programmes.
- Commencez par des programmes simples (affichage, saisie, calculs de base).

Exercices

Essayez d'écrire un programme qui :

1. Demande à l'utilisateur son nom avec `ACCEPT`.
2. Stocke le nom dans une variable.
3. Affiche "Bonjour, [nom] !" avec `DISPLAY`.

Vous trouverez sur les sites partenaires du [Pacta Cobol Network](#) des extraits en libre accès des formations payantes sur le langage Cobol.