

# Université **IBM i**

**7 novembre 2023**

IBM Innovation Studio Paris

**S08 – Modernisez et améliorez vos pratiques de développement sous IBM i avec l'IA**

13:30 - 14:30

**Philippe Bourgeois**

IBM France

*pbourgeois@fr.ibm.com*

 **infrasdufutur**

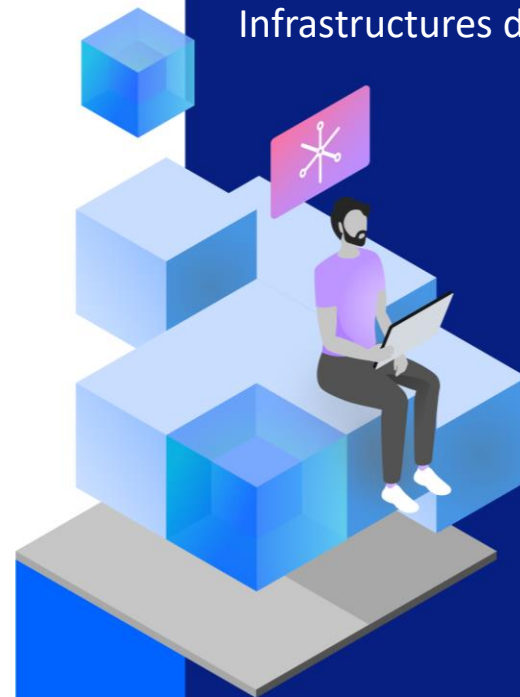
#ibmi

#uui2023

#infrastructuredufuturIBM23



Infrastructures du futur



7 et 8 novembre 2023

# Les objectifs de cette session



- Cette session a pour **objectifs** :
  - 1. De vous présenter comment l'**IA générative** peut aider les développeurs **IBM i** dans leur quotidien
    - Dans les domaines du **développement de code** et de la **modernisation d'applications IBM i**
    - Par mon retour d'expérience (entre le 29/09 et le 05/10 2023) :
      - Sur des tests d'IA générative en CL, RPG, DDS, SQL...
      - Sur les assistants au développement de code
  - 2. D'ouvrir la **discussion** et le **débat** sur ce sujet d'actualité
- **Note** : seul le thème "Développement et modernisation d'applications IBM i" sera traité. L'IA générative peut également être utilisée :
  - En IT dans l'exploitation, la sécurité, la gestion d'incidents, l'automatisation, la maintenance, le support...
  - Et bien sûr dans tous les domaines métier des différents secteurs d'activité des entreprises

# Agenda



- 1. Introduction à l'IA générative
- 2. Utilisation de l'IA générative pour les tâches de développement / modernisation sous IBM i
  - Comparaison des chatbots d'IA générative pour du code IBM i
  - Exemples
    - Génération de code
    - Débogage de code
    - Conversion de code
    - Re-engineering de code
    - Explication de code
- 3. Assistants d'IA générative dans les outils de développement (Code Assistant)
  - Les outils d'IA pour RDi et VS Code for i
  - IBM watsonx Code Assistant



**IBM i**



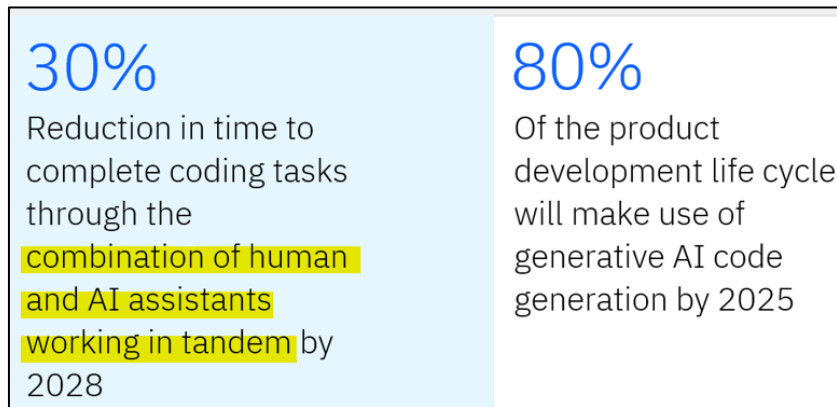
# 1. Introduction

# IA générative

- L'IA **générative** est une catégorie d'IA permettant de **générer des contenus** (textes, images, vidéos, musique, **code informatique...**), à partir de **modèles**, et en réponse à des **invites** (prompts)
- Les modèles sont de 2 types :
  - **LLM** (Large Language Model)
    - Modèles de langage de grande envergure, qui sont entraînés pour comprendre et générer du **texte** dans une langue naturelle, de manière cohérente et contextuelle
      - Exemple : GPT (Generative Pretrained Transformer) d'OpenAI
  - **FM** (Foundation Model)
    - Des LLM affinés/spécialisés pour générer différents types de données (texte, audio, images, vidéo...) et qui servent de fondation pour construire des applications d'IA

# IA générative – Génération de **code** informatique

- Une **évolution** (inexorable ?) des pratiques de développement



Source : GARTNER – Emerging Tech: Generative AI Code Assistants Are Becoming Essential to Developer Experience  
<https://www.gartner.com/en/documents/4348899>

- A aujourd'hui il n'existe **pas** encore d'outil **IBM** d'IA générative permettant d'assister les développeurs **RPG/COBOL** sur **IBM i**
  - Mais il en existe un depuis le 26/10/2023 pour l'IBM z (Watsonx Code Assistant for z)

# IA générative – Modernisation des applications

- L'IA générative peut (pourrait, pourra ?) apporter des gains de productivité dans plusieurs domaines de la **modernisation des applications**
- Par exemple, pour l'**IBM i** :

Génération de code	Conversion de code	Reverse-engineering	Autres
<ul style="list-style-type: none"> <li>• CL, RPG, COBOL, C, C++</li> <li>• Java, PHP, Python...</li> <li>• DDS (PF, LF, DSPF, PRTF)</li> <li>• SQL (DDL, DML, SQL/PSM...)</li> <li>• UI               <ul style="list-style-type: none"> <li>✓ Génération du code à partir d'images</li> <li>✓ Recommandations d'agencement/flux à partir de scénarios utilisateur</li> </ul> </li> <li>• Détection et correction de bugs</li> <li>• Complétion de code en temps réel</li> </ul>	<ul style="list-style-type: none"> <li>• DDS (PF/LF) → SQL</li> <li>• Non Unicode → Unicode</li> <li>• RPG Fixe, RPG III → RPG Full Free</li> <li>• Code L4G → RPG Full Free</li> <li>• RPG → Java, PHP, Python...</li> <li>• COBOL → Java, PHP, Python</li> <li>• DSPF → HTML/CSS/JS</li> </ul>	<ul style="list-style-type: none"> <li>• Modularisation du code               <ul style="list-style-type: none"> <li>✓ Extraction des règles métier</li> <li>✓ Monolithique vers services</li> </ul> </li> <li>• Restructuration du code</li> <li>• Code source à partir des objets               <ul style="list-style-type: none"> <li>✓ DDL à partir des objets DB2</li> <li>✓ Source RPG, COBOL... à partir de l'exécution des programmes</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• Explication du code</li> <li>• Documentation</li> <li>• Cartographie des applications</li> <li>• Création des références croisées</li> <li>• Découverte du modèle relationnel</li> <li>• Détection du code mort</li> <li>• Création de scénarios de test</li> <li>• Anonymisation de données</li> <li>• etc.</li> </ul>

## ■ 1. Chatbots

- Bard (Google)
- BingAI (Microsoft)
- ChatGPT (OpenAI)
- Chatsonic (Writesonic)
- Claude 2 (Anthropic) (US et UK)
- Grok (X) (pas encore dispo)
- HuggingChat (Hugging Face)
- Jasper (Open Source)
- Perplexity AI (Perplexity)
- YouChat (You)
- ...

## ■ 2. Assistants au développement

- Bito (Bito)
- CodeWhisperer (Amazon)
- CoPilot (GitHub et OpenAI)
- Replit (Replit)
- Tabnine (Tabnine)
- Watsonx Code Assistant (IBM)
- ...
- Intégration des chatbots dans les outils de développement



A decorative graphic on the left side of the slide. It features several blue, semi-transparent 3D cubes of varying sizes and orientations. At the bottom, there is a grey, angular platform or base. The text 'IBM i' is positioned in the middle of this graphic area.

**IBM i**

2. Utilisation de l'IA générative pour  
les tâches de développement /  
modernisation sous IBM i

# Chatbots – Le prompt (invite)

- **Prompt** : la demande que vous allez soumettre au chatbot
- Quelques **règles** :
  - **1.** Assigner un **rôle**, une identité, une profession au chatbot :
    - *Tu es un expert IBM i*
    - *Tu es un développeur en langage RPG sous IBM i*
    - *Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours \*\*FREE en première ligne*
  - **2.** Être **précis** dans la demande (fournir un maximum de contexte et de détails) :
    - *Peux-tu écrire un programme en langage RPG Full Free, sans utiliser de procédure, qui affiche "Bonjour" à l'écran avec l'instruction DSNLY ?*
    - *Plutôt que "Peux-tu écrire un programme RPG qui affiche "Bonjour" à l'écran ?"*

# Chatbots – Le prompt (invite)

- Quelques règles (suite) :
  - 3. Effectuer des requêtes à la chaîne :
    - Après avoir demandé : *Peux-tu écrire un programme en langage RPG Full Free, sans procédure, qui affiche "Bonjour" à l'écran avec l'instruction DSPLY ?*  
si le résultat obtenu n'est pas correct (parce que par exemple il manque une instruction de fin de programme), enchaîner avec une seconde requête :
      - *Ce programme ne fonctionne pas car il lui manque une instruction pour s'arrêter*
  - 4. Autres "conseils" :
    - Donner un exemple de la réponse attendue
    - Utiliser des verbes précis et à la forme affirmative
    - Reformuler, reposer la question, dialoguer
    - Utiliser des chatbots qui permettent de **conserver le contexte** pour éviter de le préciser à chaque nouvelle demande




# Comparaison des chatbots d'IA généralive pour du code IBM i



# ChatGPT 3.5 – 1<sup>er</sup> programme simple – Test 1

PB


Peux-tu créer un programme en langage IBM RPG Full Free (avec **\*\*free** en 1<sup>ère</sup> ligne), sans procédure, qui affiche "Bonjour" par DSPLY ? 

SANS contexte



Bien sûr ! Voici un exemple de programme en langage IBM RPG Full Free qui affiche "Bonjour" en utilisant l'opération DSPLY :

rpgle

 Copy code

```
**FREE
ctl-opt dftactgrp(*no);

dcl-s message char(7) inz('Bonjour');

dsply message;

*inlr = *on;
```

Ce programme utilise l'opération DSPLY pour afficher le message "Bonjour" à l'écran. La ligne ``*inlr = *on;`` indique que le programme se termine après l'exécution de cette ligne. Notez que j'ai inclus la directive ``ctl-opt dftactgrp(*no);`` pour spécifier que le programme ne sera pas lié à un groupe d'activation par défaut, ce qui est courant dans les programmes RPG Full Free.

# ChatGPT 4.0 – 1<sup>er</sup> programme simple – Test 1



PB

Peux-tu créer un programme en langage IBM RPG Full Free (avec `**free` en 1ère ligne), sans procédure, qui affiche "Bonjour" par `DSPLY` ?

SANS contexte



Le langage RPG (Report Program Generator) d'IBM est un langage de programmation principalement utilisé sur les systèmes IBM iSeries (AS/400). Dans sa version moderne, vous pouvez écrire du code en "full free" format, ce qui signifie que vous n'êtes pas limité à la grille de colonnes traditionnelle.

Voici un exemple simple de programme RPG en format "full free" qui utilise l'opération

``DSPLY`` pour afficher le message "Bonjour" :

```
rpg
**free
ctl-opt option(*srcstmt : *nodebugio);

dcl-s message char(7) inz('Bonjour');

// Affiche le message "Bonjour"
dsply message;

*inlr = *on;
```

Ce programme déclare une variable ``message`` de type ``char`` (chaîne de caractères) et initialise celle-ci avec la valeur ``'Bonjour'``. Ensuite, il utilise l'instruction ``dsply`` pour afficher le contenu de la variable ``message`` sur l'écran. Finalement, il définit ``*inlr`` à ``*on`` pour indiquer que le programme est prêt à se terminer.

Assurez-vous de sauvegarder le code source dans un membre de type RPGLE et de compiler le programme avec la commande ``CRTBNDRPG`` avant de l'exécuter.

Régé

# ChatGPT 3.5 – 1<sup>er</sup> programme simple – Test 2



PB

Tu es un expert IBM i (anciennement iSeries, AS/400) et un développeur en langage RPG sous IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free tu indiqueras toujours **\*\*FREE** en première ligne. Peux-tu créer un programme en langage IBM RPG Full Free, sans procédure, qui affiche "Bonjour" par DSPLY

AVEC contexte



Bien sûr ! Voici un exemple de programme RPG Full Free qui affiche "Bonjour" à l'instruction DSPLY :

```
rpgle
```

```
**FREE
ctl-opt dftactgrp(*no) actgrp('MYACTGRP');

dcl-s message char(7) inz('Bonjour');

dsply message;

*inlr = *on;
```

Dans ce programme RPG Full Free :

- Nous commençons par indiquer `**FREE` en première ligne pour spécifier que c'est un programme RPG Full Free.
- Ensuite, nous utilisons `ctl-opt` pour définir quelques options de compilation. `dftactgrp(*no)` indique que ce programme ne s'exécute pas dans le groupe d'activation par défaut, et `actgrp('MYACTGRP')` spécifie le nom du groupe d'activation dans lequel le programme sera exécuté. Vous pouvez personnaliser le nom du groupe d'activation en fonction de vos besoins.
- Nous déclarons une variable `message` de type caractère avec une longueur de 7 caractères et nous l'initialisons avec la chaîne "Bonjour".
- Ensuite, nous utilisons l'instruction `dsply` pour afficher le contenu de la variable `message` sur la console.
- Enfin, nous utilisons `*inlr = *on` pour indiquer que le programme se termine après son exécution.

Vous pouvez copier ce code dans un membre source RPG Full Free, le compiler et l'exécuter sur votre système IBM i pour afficher "Bonjour" à l'écran. Assurez-vous d'ajuster les options de compilation et le groupe d'activation en fonction de votre environnement.

# ChatGPT 4.0 – 1<sup>er</sup> programme simple – Test 2

PB

Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours **\*\*FREE** en première ligne. Peux-tu créer un programme en langage IBM RPG Full Free, sans procédure, qui affiche "Bonjour" par DSPLY ?

AVEC contexte



Bien sûr! En RPG Full Free, voici un exemple de programme simple qui affiche "Bonjour" à l'aide de l'op-code **DSPLY**:

```
RPGLE
```

```
**FREE
ctl-opt option(*srcstmt : *nodebugio);

dcl-s Message varchar(50);

Message = 'Bonjour';
dsply Message;

*inlr = *on;
```

Expliquons un peu le code ci-dessus :

- **\*\*FREE** en première ligne indique que le code est écrit en format Free RPG, sans colonnes fixes pour les opérations, ce qui rend le code plus lisible.
- **ctl-opt** est utilisé pour définir les options de contrôle du programme. Dans cet exemple, j'ai utilisé **\*srcstmt** pour que les numéros de ligne de l'erreur correspondent à ceux du source, et **\*nodebugio** pour empêcher l'affichage des E/S de débogage.
- **dcl-s** est utilisé pour déclarer une variable. Ici, j'ai déclaré une variable **Message** de type varchar.
- Ensuite, nous assignons la chaîne 'Bonjour' à la variable **Message**.
- L'op-code **dsply** est utilisé pour afficher le contenu de la variable **Message**.
- Enfin, **\*inlr = \*on;** est utilisé pour indiquer la fin du programme.

Une fois ce code compilé et exécuté, il affichera "Bonjour" sur l'écran.



# ChatGPT 3.5 – 1<sup>er</sup> programme simple – Test 3

PB


Peux-tu créer le même programme mais cette fois-ci en RPG IV format fixe (avec des spécifications en colonne) ?

SANS contexte



Bien sûr ! Voici le même programme en langage RPG IV au format fixe avec des spécifications en colonnes :

rpgle

 Copy code

```
**FREE  
ctl-opt   dftactgrp(*no) ;  
  
dcl-s    message   char(7)   inz('Bonjour') ;  
  
dsply   message ;  
  
*inlr = *on ;
```

Ce programme fait la même chose que le précédent en utilisant la syntaxe RPG IV au format fixe avec des spécifications en colonnes. La directive `ctl-opt` est utilisée pour spécifier les options de compilation, et le reste du code suit la structure typique d'un programme RPG IV.

# ChatGPT 3.5 – 1<sup>er</sup> programme simple – Test 3



PB

Peux-tu créer le même programme mais cette fois-ci en RPG IV format fixe (avec des spécifications en colonne) ?

AVEC contexte



Bien sûr ! Voici le même programme qui affiche "Bonjour" en utilisant le format fixe RPG IV avec des spécifications en colonnes :

rpgle

```
H DFTACTGRP(*NO) ACTGRP('MYACTGRP')
D message          S              7  INZ('Bonjour')

C      message      DSPLY

C                      SETON          LR
```

Dans ce programme RPG IV format fixe :

- La ligne H (entête) contient les spécifications du programme, telles que les options de compilation. Nous utilisons ``DFTACTGRP(*NO)`` pour indiquer que le programme ne s'exécute pas dans le groupe d'activation par défaut et ``ACTGRP('MYACTGRP')`` pour spécifier le nom du groupe d'activation.
- Ensuite, nous déclarons une variable ``message`` de type caractère avec une longueur de 7 caractères et nous l'initialisons avec la chaîne "Bonjour". Cette déclaration est effectuée dans la section D (déclarations) du programme.
- La ligne C est une spécification de calcul qui utilise l'instruction ``DSPLY`` pour afficher le contenu de la variable ``message`` sur la console.
- Enfin, la ligne C avec ``SETON LR`` indique que le programme se termine après son exécution.

Vous pouvez copier ce code dans un membre source RPG IV format fixe, le compiler et l'exécuter sur votre système IBM i pour afficher "Bonjour" à l'écran. Assurez-vous d'ajuster les options de compilation et le groupe d'activation en fonction de votre environnement.

# ChatGPT 4.0 – 1<sup>er</sup> programme simple – Test 3


PB

Peux-tu créer le même programme mais cette fois-ci en RPG IV format fixe (avec des spécifications en colonne) ?



Bien sûr, voici comment vous pouvez écrire ce programme en RPG IV avec un format fixe :

RPGLE

 Copy code

```
H option(*SRCSTMT : *NODEBUGIO)
D Message      S          50A  varying
/free
  Message = 'Bonjour';
  dsply Message;
  *inlr = *on;
/end-free
```

AVEC contexte

Dans cet exemple :

- La spécification ``H`` est utilisée pour les options de contrôle du programme, similaires à ``ctl-opt`` dans le format libre.
- La spécification ``D`` est utilisée pour déclarer la variable ``Message``.
- Les blocs ``/free`` et ``/end-free`` délimitent une section de code qui peut être écrite en format libre, même dans un programme en format fixe.
- ``Message = 'Bonjour';``, ``dsply Message;``, et ``*inlr = *on;`` font la même chose que dans l'exemple précédent.

Notez que j'ai utilisé une section de code libre entre les balises ``/free`` et ``/end-free`` car il est généralement préférable de le faire si le compilateur le prend en charge. Cela rend le code plus lisible et plus facile à maintenir. Vous pouvez aussi écrire ces opérations en format fixe, mais c'est beaucoup moins courant pour les versions modernes d'RPG IV.


# Bing – 1<sup>er</sup> programme simple – Test 2

Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours **\*\*FREE** en première ligne. Peux-tu créer un programme en langage IBM RPG Full Free, sans procédure, qui affiche "Bonjour" en utilisant le code-opération DSPLY ?

Bien sûr, voici un exemple simple de programme en langage IBM RPG Full Free qui affiche "Bonjour" en utilisant le code-opération DSPLY :

```
**FREE
DCL-S Bonjour CHAR(7) INZ('Bonjour');
DSPLY Bonjour;
*INLR = *ON;
```

Dans ce programme, nous déclarons une variable `Bonjour` de type `CHAR(7)` et nous l'initialisons avec la valeur `'Bonjour'`. Ensuite, nous utilisons le code-opération `DSPLY` pour afficher la valeur de la variable `Bonjour`. Enfin, nous mettons fin au programme en affectant `*ON` à `*INLR`.

1 sur 10 



Quelles sont les autres opérations disponibles en RPG Full Free?

Comment déclarer une variable numérique en RPG Full Free?

# Bing – 1<sup>er</sup> programme simple – Test 3 🙄

Peux-tu créer le même programme mais cette fois-ci en RPG format fixe (c'est-à-dire pas en format libre), donc avec des spécifications (par exemple la spécification D pour déclarer les variables et la spécification C pour définir les traitements) ?

Bien sûr, voici un exemple de programme en langage IBM RPG en format fixe qui affiche "Bonjour" en utilisant le code-opération DSPLY :

```
D Bonjour      S          7  INZ('Bonjour')
C              DSPLY      Bonjour
C              SETON
LR
```

Dans ce programme, nous utilisons la spécification D pour déclarer une variable `Bonjour` de type `S` (chaîne de caractères) de longueur `7` et nous l'initialisons avec la valeur `'Bonjour'`. Ensuite, nous utilisons la spécification C avec le code-opération `DSPLY` pour afficher la valeur de la variable `Bonjour`. Enfin, nous mettons fin au programme en affectant `ON` à `LR`.

# Chatsonic – 1<sup>er</sup> programme simple – Test 2



Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours **\*\*FREE** en première ligne. Peux-tu créer un programme en langage IBM RPG Full Free, sans procédure, qui affiche "Bonjour" par DSPLY ?



Bien sûr! Voici un exemple de programme en langage RPG Full Free qui affiche "Bonjour" à l'aide de l'instruction DSPLY:

```
**FREE
ctl-opt dftactgrp(*no) actgrp('MYACTGRP');

dcl-s message char(10) inz('Bonjour');

dsply message;

return;
```

Copy code

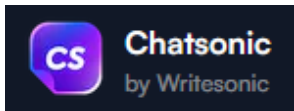
Ce programme commence par la directive **\*\*FREE** qui indique que le code est en RPG Full Free. Ensuite, l'instruction `ctl-opt` est utilisée pour définir les options du programme. Dans cet exemple, `dftactgrp(*no)` spécifie que le programme n'utilise pas le groupe d'activation par défaut et `actgrp('MYACTGRP')` spécifie le groupe d'activation à utiliser.

Ensuite, la variable `message` est déclarée comme une chaîne de caractères de 10 caractères avec la valeur initiale "Bonjour".

Enfin, l'instruction `dsply` est utilisée pour afficher le contenu de la variable `message` à l'aide de l'écran de l'IBM i.

N'oubliez pas de compiler et d'exécuter ce programme sur votre système IBM i pour voir le message "Bonjour" affiché à l'écran.

Mais le test 3 (même programme en format fixe) a échoué



# You – 1<sup>er</sup> programme simple – Test 2 – En français

Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours **\*\*FREE** en première ligne. Peux-tu créer un programme en langage IBM RPG Full Free, sans procédure, qui affiche "Bonjour" en utilisant le code-opération DSPLY ?

YOU.com

D'accord, je peux vous aider à créer un programme en langage IBM RPG Full Free qui affiche "Bonjour" en utilisant le code-opération DSPLY. Voici le code que vous pouvez utiliser :

```
**FREE  
ctl-opt dftactgrp(*no);  
  
dsply 'Bonjour';
```



Ce programme utilise le code-opération DSPLY pour afficher le message "Bonjour" à l'écran. La ligne **\*\*FREE** indique que le programme est en RPG Full Free. La directive **ctl-opt dftactgrp(\*no);** spécifie que le programme ne rejoindra pas automatiquement le groupe d'activation par défaut.

J'espère que cela vous aide ! Si vous avez d'autres questions, n'hésitez pas à les poser.

# You – 1<sup>er</sup> programme simple – Test 2 – En anglais



You're an expert on the IBM i system (formerly known as iSeries or AS/400) and a developer of RPG language on the IBM i system. You know RPG III, RPG IV, RPG Free-Format and RPG Full Free. For the RPG Full Free programs you create, you'll always indicate `**FREE` in the first line. Can you create a program in IBM RPG Full Free language, without procedures, that displays "Hello" using the `DSPLY` operation code?

Mais le test 3 (même programme en format fixe) a échoué

Certainly! I can help you create an RPG Full Free program that displays "Hello" using the `DSPLY` operation code. Here's an example program:

```
**FREE
ctl-opt dftactgrp(*no) actgrp('HELLO');

dcl-s message char(5) inz('Hello');

dsply message;

return;
```

Let's go through the code step by step:

- The `**FREE` directive at the beginning of the program indicates that it is written in RPG Full Free format.
- The `ctl-opt` statement sets the default activation group to `*NO`, and also specifies the activation group to be used as `'HELLO'`.
- The `dcl-s` statement declares a character variable named `message` and initializes it with the value `'Hello'`.
- The `dsply` operation code is used to display the value of the `message` variable on the screen.
- Finally, the `return` statement is used to end the program.

This program will display "Hello" on the screen when executed.



# Perplexity AI – 1<sup>er</sup> programme simple – Test 2



Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours **\*\*FREE** en première ligne. Peux-tu créer un programme en langage IBM RPG Full Free, sans utiliser de procédure, qui affiche "Bonjour" en utilisant le code-opération DSPLY ?

Mais le test 3 (même programme en format fixe) a échoué

## ☰ Sources

<a href="#">RPG Talks To Watson - IT Jungle</a> itjungle · 1	<a href="#">ODW - Multi-Language System Plugin   RPG...</a> rpgmakerweb · 2	<a href="#">[PDF] AS/400 Developer Kit for Java - CiteSeerX</a> psu · 3
<a href="#">Homepage - Cookbook   Mbed</a> mbed · 5	<a href="#">Sound Show by Impronivers</a> itch · 6	

Voici un exemple de programme en langage IBM RPG Full Free qui affiche "Bonjour" en utilisant le code-opération DSPLY, sans utiliser de procédure :


```
**FREE  
ctl-opt main(Main);  
  
dcl-s message char(7) inz('Bonjour');  
  
dsply message;  
  
return;
```


Ce programme commence par la directive **\*\*FREE**, qui indique que le code est en format libre. Ensuite, il y a la spécification de l'option de contrôle main, qui indique que le point d'entrée du programme est la procédure Main. La variable message est déclarée comme une chaîne de caractères de longueur 7, initialisée à la valeur "Bonjour". Enfin, le code-opération DSPLY est utilisé pour afficher le contenu de la variable message à l'écran. Le programme se termine par l'instruction return.



# Bard – 1<sup>er</sup> programme simple – Test 2

**P** Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours **\*\*FREE** en première ligne. Peux-tu créer un programme en langage IBM RPG Full Free, sans procédure, qui affiche "Bonjour" par DSPLY ?

Afficher d'autres suggestions 

 Bien sûr, voici un programme en langage IBM RPG Full Free, sans procédure, qui affiche "Bonjour" par DSPLY :

## Extrait de code

```
**FREE

// Déclarations
D txt *STRING(10)

// Initialisation
txt = "Bonjour"

// Affichage
DSPLY txt
```

Ce programme utilise les éléments suivants :

- La directive **\*\*FREE** en première ligne indique qu'il s'agit d'un programme en RPG Full Free.
- La déclaration `D txt *STRING(10)` déclare une variable `txt` de type `STRING` d'une longueur maximale de 10 caractères.
- L'instruction `txt = "Bonjour"` initialise la variable `txt` avec la chaîne de caractères "Bonjour".
- L'instruction `DSPLY txt` affiche la variable `txt` sur la console.

Utilisez le code avec précaution. [En savoir plus](#)

Idem si la  
demande  
est en  
anglais

Bard Expérimental

# HuggingChat – 1<sup>er</sup> programme simple – Test 2 🙄

Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours `**FREE` en première ligne. Peux-tu créer un programme en langage IBM RPG Full Free, sans procédure, qui affiche "Bonjour" par DSPLY ?

Voici un exemple de programme en RPG Full Free qui affiche "Bonjour" par DSPLY :

```
FREE  
DSPLY 'Bonjour'
```

Explications :

- ``FREE`` est la première ligne du programme, qui indique que ceci est un programme RPG Full Free.
- ``DSPLY`` est une instruction de display (affichage) qui permet d'afficher un message sur l'écran.
- ``'Bonjour'`` est le message à afficher.

Notez que cette syntaxe est spécifique au RPG Full Free, et ne fonctionnera pas avec les versions antérieures du RPG (RPG III, RPG IV, etc.).

Idem si la  
demande  
est en  
anglais

# Pour la suite des tests

- Pour les tests qui vont suivre j'ai utilisé **ChatGPT 4** et j'ai sauvegardé le contexte global via la fonction "Instructions personnalisées"

Instructions personnalisées ⓘ

Que souhaiteriez-vous que ChatGPT sache sur vous pour fournir de meilleures réponses ?

Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours **\*\*FREE** en première ligne. Tu connais également le langage DDS (Data Description Specifications - Spécifications de Description des Données) sur le système IBM i, qui permet de créer des fichiers physiques (PF - Physical File), des

883/1500 Masquer les conseils ⓘ

Tu es un expert sur le système IBM i (appelé anciennement iSeries ou AS/400). Tu es également un développeur en langage RPG sur le système IBM i. Tu connais le RPG III, le RPG IV, le RPG Free-Format et le RPG Full Free. Pour les programmes en RPG Full Free que tu vas créer, tu indiqueras toujours **\*\*FREE** en première ligne. Tu connais également le langage DDS (Data Description Specifications - Spécifications de Description des Données) sur le système IBM i, qui permet de créer des fichiers physiques (PF - Physical File), des fichiers logiques (LF - Logical File), des fichiers écran (DSPF - Display File) et des fichiers imprimante (PRTF - Printer File). Tu connais également le langage CL (Control Language) sur le système IBM i, ainsi que le langage SQL sur le système IBM i qui permet de créer et gérer des fichiers de la base de données DB2 for i (appelée également DB2/400).

- Mon plan
- Instructions personnalisées
- Paramètres & Beta
- Se déconnecter

PB pbourgeois@fr.ibm.cor ...



# Génération de code



# RPG Full Free – Lecture d'un fichier DB2 (natif)

PB

Tu as une table FILMS avec 2 colonnes (NUMERO\_FILM de type INTEGER et TITRE\_FILM de type VARCHAR(50)). Peux-tu créer un programme qui affiche à l'écran par DSPLY la colonne TITRE\_FILM de tous les films de la table FILMS ? La table FILMS pourra avoir été créée en DDS ou en SQL. Le programme devra être en RPG Full Free et ne devra pas contenir de SQL



```
rpg

**FREE
dcl-f FILMS usage(*input);

dcl-s titreFilm varchar(50);

read(e) FILMS;
dow (not %eof(FILMS));
  titreFilm = TITRE_FILM;
  dsply titreFilm;
  read(e) FILMS;
enddo;

*inlr = *on;
```

# RPG Full Free – Lecture d'un fichier DB2 (SQL)

PB

Peux-tu créer le même programme mais avec des instructions SQL imbriquées dans le programme RPG (programme de type SQLRPGLE)?



```
rpg

**FREE
dcl-s titreFilm varchar(50);

exec sql DECLARE C1 CURSOR FOR
      SELECT TITRE_FILM FROM FILMS;
exec sql OPEN C1;

do (%sqlstt = '02000');
  exec sql FETCH C1 INTO :titreFilm;
  if (%sqlstt = '00000');
    dsply titreFilm;
  endif;
enddo;

exec sql CLOSE C1;

*inlr = *on;
```

# RPG Fixe – Lecture d'un fichier DB2 (natif)

**PB** Tu as une table FILMS avec 2 colonnes (NUMERO\_FILM de type INTEGER et TITRE\_FILM de type VARCHAR(50)). Peux-tu créer un programme qui affiche à l'écran par DSPLY la colonne TITRE\_FILM de tous les films de la table FILMS ? La table FILMS pourra avoir été créée en DDS ou en SQL. Le programme devra être en RPG avec des spécifications colonnées et non pas en RPG Full Free et ne devra pas contenir de SQL



```
rpg
H DECEDIT('0,') DATEDIT(*DMY/)
*****
F FILMS      IF  E          K DISK
*****
D TITLE      S          50A
*****
C           read(e) FILMS
C           dow not %eof(FILMS)
C           eval      TITLE = TITRE_FILM
C           dsply          TITLE
C           read(e) FILMS
C           enddo
C           Manque EVAL → *inlr = *on
C           return
```



# RPG Fixe – Lecture d'un fichier DB2 (SQL)

PB

Peux-tu créer le même programme mais avec des instructions SQL imbriquées dans le programme RPG (programme de type SQLRPGLE) ?



```

rpg
Copy code

H DECEdit('0,') DATEDIT(*DMY/) ACTGRP(*NEW)
*****
D TITLE        S          50A  varying
*****
D STMT         S          256A  inz('SELECT TITRE_FILM FROM FILMS')
D SQLCODE      S          10I  0
*****
C/exec sql DECLARE C1 CURSOR FOR S1
C/END-EXEC
*****
C/exec sql PREPARE S1 FROM :STMT
C/END-EXEC
*****
C/exec sql OPEN C1
C/END-EXEC
*****
C                    eval      SQLCODE = 0
C/exec sql FETCH NEXT FROM C1 INTO :TITLE
C/END-EXEC
C                    dow      SQLCODE = 0
C                    dsply          TITLE
C/exec sql FETCH NEXT FROM C1 INTO :TITLE
C/END-EXEC
C                    enddo
*****
C/exec sql CLOSE C1
C/END-EXEC
*****
C                    *inlr = *on
C                    return

```

# RPG Full Free – Génération de logique (1/2)

PB

Tu vas écrire un programme RPG Full Free. Ce programme va déclarer la variable JSONVAR de type VARCHAR(100) et initialisée à '{"MATRIX", "SPIDERMAN", "INCEPTION", "GLADIATOR"}'. Ce programme devra décomposer le contenu de cette variable JSONVAR afin d'afficher à l'écran par DSPLY les valeurs qui sont contenues dans cette variable JSONVAR (à savoir : MATRIX, SPIDERMAN, INCEPTION et GLADIATOR).



Bien sûr, voici un exemple de code en RPG Full Free qui fait ce que vous avez demandé. Veuillez noter que dans RPG Free Format, il n'y a pas de bibliothèque JSON standard intégrée. Par conséquent, cette démonstration est très simplifiée et spécifique au format de la chaîne donnée.

# RPG Full Free – Génération de logique (2/2)

```
rpg
Copy code

**FREE
Dcl-S JSONVar Varchar(100) Inz('{"MATRIX", "SPIDERMAN", "INCEPTION", "GLADIA
Dcl-S StartPos Packed(3);
Dcl-S EndPos Packed(3);
Dcl-S JSONElement Varchar(50);
Dcl-S JSONLength Packed(3);

// Supprimer les accolades en début et en fin de la chaîne JSON
JSONVar = %Subst(JSONVar: 2 : %Len(JSONVar) - 2);
JSONLength = %Len(%TrimR(JSONVar));

StartPos = 1;

Dow (StartPos <= JSONLength);

// Trouver la position du prochain élément dans la chaîne JSON
EndPos = %Scan("'", JSONVar: StartPos) - 1;
```

: StartPos + 1

```
: StartPos + 1

// Extraire l'élément JSON actuel
JSONElement = %Subst(JSONVar: StartPos: EndPos - StartPos + 1);

// Afficher l'élément à l'aide de DSPLY
Dsply JSONElement;

// Rechercher la position du prochain élément
StartPos = %Scan(',', JSONVar: EndPos) + 1;
If (StartPos = 1);
  Leave;
EndIf;

// Passer le guillemet ouvrant du prochain élément
StartPos += 1;

EndDo;
```

Manque return ou \*inlr = \*on

# RPG Full Free – Génération de logique (1/2)

PB

Tu as une table PALINS avec une colonne MOT de type VARCHAR(50). Cette table PALINS contient les données suivantes :

radar

watson

ressasser

repasser

kayak

Peux-tu créer un programme qui affiche à l'écran par DSPLY la colonne MOT de la table PALINS uniquement si la colonne MOT contient un palindrome ? La table PALINS pourra avoir été créée en DDS ou en SQL. Le programme devra être en RPG Full Free et ne devra pas contenir de SQL

# RPG Full Free – Génération de logique (2/2)

RPG

```
**FREE

// Déclaration du fichier PALINS
dcl-f PALINS keyed disk;

dcl-s i INT(5);
dcl-s j INT(5);
dcl-s len INT(5);
dcl-s isPalindrome CHAR(1) INZ('Y');

// Lire le fichier PALINS
read(e) PALINS;
dow not %eof;

// Réinitialiser le flag de palindrome
isPalindrome = 'Y';

// Obtenir la longueur du mot
len = %len(%trimr(MOT));
```

```
// Boucle pour vérifier si le mot est un palindrome
for i = 1 to len / 2; → %int(len / 2)
  j = len - i + 1;
  if %subst(MOT: i: 1) <> %subst(MOT: j: 1);
    isPalindrome = 'N';
    leave;
  endif;
endfor;

// Afficher le mot si c'est un palindrome
if isPalindrome = 'Y';
  dsply MOT;
endif;

// Lire le prochain enregistrement
read(e) PALINS;
enddo;

*INLR = *ON;
```

```
DSPLY  radar
DSPLY  ressasser
DSPLY  kayak
```

# PF et LF

PB

Peux-tu générer la source DDS d'un fichier physique FILMS avec une zone CODE de type numérique étendu de longueur 6, une zone TITRE alphanumérique de longueur 40 et une zone ANNEE de type numérique étendu de longueur 4. Une clé sera définie sur CODE en UNIQUE



Bien sûr, voici un exemple de source DDS pour créer un fichier physique **FILMS** avec les spécifications que vous avez fournies.

```
dds Copy code
A      R FILMSR
A      CODE      6P 0      COLHDG('Code du film')
A      TITRE     40A      COLHDG('Titre du film')
A      ANNEE     4P 0      COLHDG('Année du film')
A      K CODE
A      UNIQUE
```

PB

Dans les DDS le mot-clé UNIQUE est indiqué avant le nom du format (enregistrement)  
Et le type numérique étendu c'est S et non P

```
R FILMSR      PFILE (FILMS)
K ANNEE
```

PB

Peux-tu générer la source d'un fichier logique (LF) qui s'appuie sur le PF FILMS avec une clé sur la zone ANNEE ?

```
R FILMSR      UNIQUE
CODE          6S 0      COLHDG('Code du film')
TITRE         40A      COLHDG('Titre du film')
ANNEE         4S 0      COLHDG('Année du film')
K CODE
```

PB

Tu as une table FILMS avec 2 colonnes (CODE de type numérique de longueur 6 et TITRE de type alphanumérique de longueur 40 ). Peux-tu créer les DDS d'un fichier écran (DSPF) avec un sous-fichier (SFL) de 10 lignes qui affichera al zone CODE en ligne 15, colonne 5 et la zone TITRE en ligne 5, colonne 25 ?

```
dds Copy code
A          DSPSIZ(24 80 *DS3)
A          PRINT
A          CA03(03 'Exit')
A          INDARA
A          R SFLRECORD          SFL
A          CODE          6S 00 15 5
A          TITRE          40A 0 5 25
A          R SFLCTL          SFLCTL(SFLRECORD)
A          OVERLAY
A          SFLPAG(0010)
A          SFLSIZ(0011)
A N38          SFLDSP
A N38          SFLDSPCTL
A 38          SFLEND(*MORE)
A          SFLCLR
A          SFLRRN          4S 0H
```

# DSPF – Génération de DDS à partir d'une image



PB

Application VIDEO - Recherche de films

Copyright IBM France - 2013


1. Recherche par titre de film  
2. Recherche par nom d'acteur

F3=Exit F12=Annuler

Peux-tu générer le code DDS de ce DSPF ?



# DSPF – Génération de DDS à partir d'une image

 Bien sûr, je peux vous aider à créer une ébauche du DDS (Data Description Specifications) pour le DSPF (Display File) en fonction de l'image que vous m'avez fournie. Cela ressemblerait à quelque chose comme ceci:

```

DDS
A          DSPSIZ(24 80 *DS3)
A          INDARA
A          PRINT
A          ERRSFL
A          R SCREEN01
A          OVERLAY
A          HEADER      2A  B  1  2'Application VIDEO - Recherche d
A          COPYRIGHT   2A  B  2  2'Copyright IBM France - 2013'
A          4  2'Recherche par titre de film'
A          5  2'Recherche par nom d'acteur'
A          OPTION      1A  B 10  2
A 10          DSPATR(HI)
A          F3          1A  B 23  2'F3=Exit'
A          F12         1A  B 24  2'F12=Annuler'
    
```

Cette ébauche de DDS est basée sur l'apparence de l'écran que vous m'avez montré. Cependant, il pourrait y avoir des fonctionnalités supplémentaires ou des modifications nécessaires pour que cela corresponde exactement à votre système ou à vos besoins. Vous devrez également ajouter la logique nécessaire pour traiter les options et les touches de fonction dans le programme qui utilise ce DSPF.

PB

Peux-tu générer un programme en langage CLLE qui envoie un message à l'utilisateur qui exécute le programme ? Ce message devra afficher la date et l'heure du système ainsi que le nom de l'utilisateur.

```
clle Copy code

PGM

DCL VAR(&DATE) TYPE(*CHAR) LEN(10)
DCL VAR(&TIME) TYPE(*CHAR) LEN(8)
DCL VAR(&USER) TYPE(*CHAR) LEN(10)
DCL VAR(&MESSAGE) TYPE(*CHAR) LEN(50)

RTVSYSVAL  SYSVAL(QDATE) RTNVAR(&DATE)
RTVSYSVAL  SYSVAL(QTIME) RTNVAR(&TIME)
RTVJOBA    USER(&USER)

CHGVAR VAR(&MESSAGE) VALUE('Date: ' || &DATE || ' Time: ' || &TIME
                             ' User: ' || &USER)

SENDPGMMSG MSGID(CPF9898) MSGF(QCPFMSG) MSGDTA(&MESSAGE) TOPGMQ(*EX)

ENDPGM
```

PB

Peux-tu générer le source SQL de la création d'une table DB2 for i avec les caractéristiques suivantes :

- Nom de la table : FILMS
- Nom du format (enregistrement) : FFILMS
- Nom et attributs des colonnes :
  - i. CODE de type INTEGER, de type IDENTITY
  - ii. TITRE de type VARCHAR(50)
  - iii. ANNEE de type numérique de longueur 4
- CODE sera la clé primaire



Bien sûr, voici le code SQL pour créer la table `FILMS` avec les spécifications que vous avez fournies sur un système IBM i avec DB2 for i :

```
sql Copy code  
  
CREATE TABLE FILMS (  
    CODE INTEGER GENERATED BY DEFAULT AS IDENTITY (START WITH 1, INCREMENT BY 1),  
    TITRE VARCHAR(50),  
    ANNEE NUMERIC(4, 0),  
    PRIMARY KEY (CODE)  
) RCFMT FFILMS;
```


PB

Tu as une table FILMS avec 2 colonnes (CODE de type numérique de longueur 6 et TITRE de type alphanumérique de longueur 40). Peux-tu créer une fonction table (UDTF) en SQL pour DB2 for i, qui prendra CODE en paramètre d'entrée et qui renverra une table constituée de TITRE ?



Bien sûr, voici comment créer une fonction de table utilisateur (UDTF) en SQL pour DB2 for i qui prend `CODE` comme paramètre d'entrée et renvoie une table avec la colonne `TITRE`.

sql

 Copy code

```
CREATE FUNCTION GET_TITRES_BY_CODE (P_CODE NUMERIC(6))
RETURNS TABLE (TITRE VARCHAR(40))
LANGUAGE SQL
READS SQL DATA
BEGIN
    RETURN
    SELECT TITRE FROM FILMS WHERE CODE = P_CODE;
END;
```



# Débogage de code



# RPG Full Free – Exemple 1

PB

Peux-tu corriger le programme suivant qui ne fonctionne pas :

```
**free
```

```
read films;  
dow not %eof;  
z1 = titre;  
dsply z1;  
enddo;
```

```
*inlr = *on;
```

Dans cet exemple, j'ai supposé que ``films`` est un fichier disque et ``titre`` est un champ dans ce fichier. J'ai également ajouté le code ``(e)`` après la commande ``read`` pour gérer les erreurs sans planter le programme. La variable ``z1`` est déclarée comme une chaîne de caractères de longueur 50, ce qui pourrait être ajusté en fonction de la longueur réelle de votre champ ``titre``.

N'oubliez pas d'adapter ces parties du code selon votre cas réel.



Bien sûr, je vois quelques points qui nécessitent des corrections dans votre programme RPG Full Free.

1. Vous avez utilisé ``**free`` ce qui est correct pour indiquer qu'il s'agit d'un programme RPG Full Free.
2. Le fichier ``films`` n'est pas défini.
3. Vous utilisez ``titre`` mais il n'est pas défini non plus.
4. ``z1`` n'est pas défini.
5. ``not %eof`` devrait être ``%eof(films)`` pour tester la fin de fichier sur ``films``.
6. La directive ``dsply`` devrait utiliser une variable de type caractère.

Voici un exemple corrigé :

```
rpg  
  
**free  
dcl-f films disk;  
dcl-s z1 char(50);  
  
read(e) films;  
dow not %eof(films);  
z1 = titre;  
dsply z1;  
read(e) films;  
enddo;  
  
*inlr = *on;
```

Copy code

# RPG Full Free – Exemple 2

PB

Peux-tu corriger le programme suivant qui ne fonctionne pas :

```
**free
dcl-s z1a char(8) inz('21.06.23');
test(de) *dmy z1a;
if %error;
  dsply 'Erreur : Z1A n'est pas au format JOUR.MOIS.ANNEE';
else;
  dsply 'Z1A est au format JOUR.MOIS.ANNEE';
endif;
return;
```



Bien sûr. Le problème est que `TEST(DE)` s'attend à un format de date sans séparateurs. Les séparateurs, comme les points dans `'21.06.23'`, ne seront pas traités correctement. Une solution serait de retirer les points avant de faire le test.

Voici comment vous pourriez ajuster le code :

```
rpg Copy code

**free
dcl-s z1a char(8) inz('21.06.23');
dcl-s z1b char(6);

// Enlever les points
z1b = %subst(z1a: 1: 2) + %subst(z1a: 4: 2) + %subst(z1a: 7: 2);

// Tester la date
test(de) *dmy z1b;

if %error;
  dsply 'Erreur : Z1A n'est pas au format JOUR.MOIS.ANNEE';
else;
  dsply 'Z1A est au format JOUR.MOIS.ANNEE';
endif;

return;
```

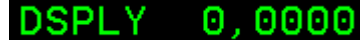
# RPG Full Free – Exemple 3 (1/2)

PB

Le programme suivant ne fonctionne pas. Le code-opération DSPLY affiche 0,0000 alors qu'il devrait afficher 0,3333. Peux-tu le corriger ?

```
ctl-opt actgrp(*new) decedit('0,');
```

```
dsply %char(proc1(1:100:0:3));
```



```
return ;
```

```
dcl-proc proc1;
```

```
  dcl-pi *n zoned(15:4);
```

```
    z1 zoned(15:2) const;
```

```
    z2 zoned(5:2) const;
```

```
    z3 zoned(9:4) const;
```

```
    z4 zoned(9:4) const;
```

```
  end-pi;
```

```
  return z1/(((z2/100)*z3)+z4);
```

```
end-proc ;
```



# RPG Full Free – Exemple 3 (1/2)

Le problème réside dans la façon dont RPG gère la division avec des opérandes entiers. Dans votre cas, la partie ``(((z2/100)*z3)+z4)`` sera évaluée comme une valeur entière, ce qui engendrera une perte de précision lors de la division.

Pour corriger cela, vous pouvez forcer une conversion en nombre à virgule flottante pour garantir que la division est effectuée correctement. Vous pouvez utiliser la fonction ``%FLOAT`` pour cela.

Voici une version corrigée de votre programme :

```
rpg                                                                    Copy code

ctl-opt actgrp(*new) decedit('0,');

dsply %char(proc1( 1 : 100 : 0 : 3 )) ;                               DSPLY  0,3333

return ;

dcl-proc proc1 ;
  dcl-pi *n zoned(15:4) ;
    z1 zoned(15:2) const ;
    z2 zoned(5:2) const ;
    z3 zoned(9:4) const ;
    z4 zoned(9:4) const ;
  end-pi ;

  return %float(z1)/(%float(((z2/100)*z3)+z4));
end-proc ;
```



# Conversion de code



# DDS vers SQL (1/2)

PB

Peux-tu convertir les DDS du fichier physique FILMS suivant en code SQL pour DB2 for i :

```

A                UNIQUE
A  R FILMSR
A  CODE    6S 0  COLHDG('Code du film')
A  TITRE   40A  COLHDG('Titre du film')
A  ANNEE   4S 0  COLHDG('Année du film')
A  K CODE
  
```



Bien sûr ! Le fichier physique FILMS en DDS que vous avez fourni peut être converti en une instruction SQL `CREATE TABLE` pour DB2 for i. Voici comment cela pourrait ressembler :

sql

Copy code

```

CREATE TABLE FILMS (
  CODE INTEGER NOT NULL UNIQUE,
  TITRE VARCHAR(40),
  ANNEE INTEGER,
  PRIMARY KEY (CODE)
);
  
```

PB

Il manque le nom du format/enregistrement (FILMSR) et les entêtes de colonne qui peuvent être générées en SQL par LABEL ON

## DDS vers SQL (2/2)



Vous avez raison, j'ai omis ces détails. Si vous souhaitez inclure le nom du format de l'enregistrement (FILMSR) et les en-têtes de colonne (via LABEL ON), voici une version plus complète du code SQL pour DB2 for i :

```
sql Copy code  
  
-- Création de la table  
CREATE TABLE FILMS (  
  CODE INTEGER NOT NULL UNIQUE,  
  TITRE VARCHAR(40),  
  ANNEE INTEGER,  
  PRIMARY KEY (CODE)  
) RCD_FMT FILMSR;  
  
-- Ajout des labels (en-têtes de colonne)  
LABEL ON COLUMN FILMS (  
  CODE IS 'Code du film',  
  TITRE IS 'Titre du film',  
  ANNEE IS 'Année du film'  
);
```

# RPG Fixe vers RPG Full Free

**PB** Peux-tu convertir le code RPG suivant qui contient des spécification colonnées en RPG Full Free, en gardant les mêmes instructions :

```

F FILMS  IF E    K DISK

D TITLE      S    50A

C          read  FILMS
C          dow   not %eof(FILMS)
C          eval  TITLE = TITRE_FILM
C  TITLE    dsply
C          read  FILMS
C          enddo
C          *inlr = *on
  
```

```

rpg

**FREE
dcl-f FILMS keyed disk;

dcl-s TITLE char(50);

read(e) FILMS;
dow not %eof(FILMS);
  TITLE = TITRE_FILM;
  dsply TITLE;
  read(e) FILMS;
enddo;

*inlr = *on;
  
```

# RPG Full Free vers Java (1/2)

PB

Peux-tu convertir le code RPG Full Free suivant en Java pour IBM i :

```
**FREE  
dcl-f FILMS keyed disk;  
dcl-s TITLE char(50);  
read FILMS;  
dow not %eof(FILMS);  
  TITLE = TITRE_FILM;  
  dsply TITLE;  
  read FILMS;  
enddo;  
*inlr = *on;
```

# RPG Full Free vers Java (2/2)



Bien sûr, voici une conversion possible de votre code RPG Full Free en Java pour IBM i, en utilisant les bibliothèques JTOpen pour accéder aux fichiers sur IBM i.

```
java Copy code

import com.ibm.as400.access.*;

public class ReadFilms {
    public static void main(String[] args) {
        try {
            // Connexion à IBM i
            AS400 as400 = new AS400("MON_SERVEUR", "MON_UTILISATEUR", "MON_M

            // Ouverture du fichier FILMS
            KeyedFile keyedFile = new KeyedFile(as400, "/QSYS.LIB/MA_LIB.LIB
            keyedFile.open();

            // Définition du record format
            RecordFormat recordFormat = keyedFile.getRecordFormat();

            // Lecture du premier enregistrement
            Record record = keyedFile.readFirst();
```

```
        // Boucle jusqu'à la fin du fichier
        while (record != null) {
            String TITLE = (String) record.getField("TITRE_FILM");

            // Afficher le titre
            System.out.println("Titre: " + TITLE);

            // Lecture de l'enregistrement suivant
            record = keyedFile.readNext();
        }

        // Fermeture du fichier et de la connexion
        keyedFile.close();
        as400.disconnectAllServices();

    } catch (Exception e) {
        e.printStackTrace();
    }
}
}
```

# RPG Full Free vers Python (1/2)

PB

Peux-tu convertir le code RPG Full Free suivant en Python pour IBM i :

```
**FREE  
dcl-f FILMS keyed disk;  
dcl-s TITLE char(50);  
read FILMS;  
dow not %eof(FILMS);  
  TITLE = TITRE_FILM;  
  dsply TITLE;  
  read FILMS;  
enddo;  
*inlr = *on;
```



# RPG Full Free vers Python (2/2)



Bien sûr, voici comment le code RPG Full Free peut être converti en Python en utilisant la bibliothèque `ibm_db_dbi` pour accéder à IBM i.

```
python Copy code  
  
import ibm_db_dbi  
  
def main():  
    try:  
        # Connexion à IBM i  
        conn = ibm_db_dbi.connect("DATABASE=MA_DB;HOSTNAME=mon_serveur;PORT=")  
  
        # Création d'un objet "curseur" pour exécuter des commandes SQL  
        cursor = conn.cursor()  
  
        # Exécution d'une requête pour lire les données du fichier FILMS  
        sql = "SELECT TITRE_FILM FROM MA_LIB.FILMS"  
        cursor.execute(sql)
```

```
# Exécution d'une requête pour lire les données du fichier FILMS  
sql = "SELECT TITRE_FILM FROM MA_LIB.FILMS"  
cursor.execute(sql)  
  
# Boucle pour lire et afficher les enregistrements  
while True:  
    row = cursor.fetchone()  
    if row is None:  
        break  
    TITLE = row[0]  
    print("Titre:", TITLE)  
  
# Fermeture de la connexion  
conn.close()  
  
except Exception as e:  
    print("Erreur:", e)  
  
if __name__ == "__main__":  
    main()
```



# Re-engineering de code



# Renommage de variables

PB

Peux-tu modifier ce code afin de donner un nom plus explicite aux variables :

```
**FREE
```

```
// Déclaration du fichier PALINS
```

```
dcl-f PALINS disk keyed;
```

```
dcl-s i INT(5);
```

```
dcl-s j INT(5);
```

```
dcl-s len INT(5);
```

```
dcl-s isPalindrome CHAR(1) INZ('Y'); . . .
```



Bien sûr, voici une version de votre code avec des noms de variables plus explicites :

```
RPGLE
```

```
**FREE
```

```
// Déclaration du fichier PALINS
```

```
dcl-f PALINS disk keyed;
```

```
dcl-s loopIndex INT(5);
```

```
dcl-s reverseIndex INT(5);
```

```
dcl-s wordLength INT(5);
```

```
dcl-s flagIsPalindrome CHAR(1) INZ('Y');
```

```
// Lire le fichier PALINS
```

Dans ce code, j'ai changé les noms des variables pour qu'ils soient plus descriptifs :

- ``i`` est devenu ``loopIndex`` pour indiquer qu'il s'agit de l'indice de la boucle.
- ``j`` est devenu ``reverseIndex`` pour indiquer qu'il s'agit de l'indice de caractère opposé dans le mot.
- ``len`` est devenu ``wordLength`` pour indiquer clairement qu'il s'agit de la longueur du mot.
- ``isPalindrome`` est devenu ``flagIsPalindrome`` pour indiquer qu'il s'agit d'un drapeau (flag) indiquant si le mot est un palindrome ou non.

J'espère que ces modifications rendent le code plus lisible et compréhensible.

# Restructuration du code (1/2)

PB

Est-ce que tu peux modifier ce programme afin que le code qui vérifie que la variable MOT est bien un palindrome soit **encapsulé dans une procédure** RPG ?

```
RPG
**FREE

// Déclaration du fichier PALINS
dcl-f PALINS disk keyed;

dcl-s i INT(5);
dcl-s j INT(5);
dcl-s len INT(5);
dcl-s isPalindrome CHAR(1) INZ('Y');

// Lire le fichier PALINS
read(e) PALINS;
dow not %eof;

// Réinitialiser le flag de palindrome
isPalindrome = 'Y';

// Obtenir la longueur du mot
len = %len(%trimr(MOT));
```

```
// Boucle pour vérifier si le mot est un palindrome
for i = 1 to %int(len / 2);
  j = len - i + 1;
  if %subst(MOT: i: 1) <> %subst(MOT: j: 1);
    isPalindrome = 'N';
    leave;
  endif;
endfor;

// Afficher le mot si c'est un palindrome
if isPalindrome = 'Y';
  dsply MOT;
endif;

// Lire le prochain enregistrement
read(e) PALINS;
enddo;

*INLR = *ON;
```



Bien sûr, il est bon de modulariser le code en utilisant des procédures pour une meilleure réutilisation et lisibilité. Voici comment le programme pourrait être modifié pour encapsuler la logique de vérification de palindrome dans une procédure RPG :

# Restructuration du code (2/2)

```
RPG

**FREE
ctl-opt dftactgrp(*no);

// Déclaration du fichier PALINS
dcl-f PALINS disk keyed;

// Prototypes
dcl-pr IsPalindrome ind;
    testStr VARCHAR(50) const;
end-pr;

// Lire le fichier PALINS
read(e) PALINS;
dow not %eof;

// Afficher le mot si c'est un palindrome
if IsPalindrome(MOT);
    dsply MOT;
endif;

// Lire le prochain enregistrement
read(e) PALINS;
enddo;

*INLR = *ON;
```

```
// Définition de la procédure
dcl-proc IsPalindrome;
    dcl-pi *n ind;
        testStr VARCHAR(50) const;
    end-pi;

    dcl-s i INT(5);
    dcl-s j INT(5);
    dcl-s len INT(5);

    // Obtenir la longueur de la chaîne
    len = %len(%trimr(testStr));

    // Boucle pour vérifier si la chaîne est un palindrome
    for i = 1 to %int(len / 2);
        j = len - i + 1;
        if %subst(testStr: i: 1) <> %subst(testStr: j: 1);
            return *OFF;
        endif;
    endfor;

    return *ON;
end-proc;
```

```
DSPLY radar
DSPLY ressasser
DSPLY kayak
```



# Explication de code



# Explication de code (1/2)

PB

Peux-tu expliquer ce code :

```
dcl-proc IsPalindrome;  
dcl-pi *n ind;  
  testStr VARCHAR(50) const;  
end-pi;
```

```
dcl-s i INT(5);  
dcl-s j INT(5);  
dcl-s len INT(5);
```

```
// Obtenir la longueur de la chaîne  
len = %len(%trimr(testStr));
```

```
// Boucle pour vérifier si la chaîne est un palindrome  
for i = 1 to %int(len / 2);  
  j = len - i + 1;  
  if %subst(testStr: i: 1) <> %subst(testStr: j: 1);  
    return *OFF;  
  endif;  
endfor;
```

```
return *ON;  
end-proc;
```



Ce code est un exemple de procédure RPG Free-Format sur le système IBM i pour vérifier si une chaîne de caractères donnée est un palindrome ou non. Un palindrome est un mot, une phrase ou une autre séquence de caractères qui se lit de la même manière à l'endroit et à l'envers (en ignorant les espaces, la ponctuation et la capitalisation).

Voici une explication détaillée du code :

## Déclaration de la procédure et du prototype d'interface

rpg

Copy code

```
dcl-proc IsPalindrome;  
dcl-pi *n ind;  
  testStr VARCHAR(50) const;  
end-pi;
```

- ``dcl-proc IsPalindrome;`` : Déclare le début de la procédure nommée ``IsPalindrome``.
- ``dcl-pi *n ind;`` et ``end-pi;`` : Définit le prototype de l'interface de la procédure. Ici, la procédure retourne un indicateur (``ind``) qui est un type de booléen sur IBM i.
- ``testStr VARCHAR(50) const;`` : Déclare un paramètre d'entrée ``testStr`` de type ``VARCHAR(50)`` qui est constant.

# Explication de code (2/2)

## Variables locales

```
rpg
dcl-s i INT(5);
dcl-s j INT(5);
dcl-s len INT(5);
```

- `dcl-s` déclare des variables scalaires.
- `i`, `j`, et `len` sont des entiers de 5 chiffres.

## Logique principale

```
rpg
len = %len(%trimr(testStr));
```

- Obtient la longueur de `testStr` après avoir supprimé les espaces à droite.

```
rpg
Copy code

for i = 1 to %int(len / 2);
  j = len - i + 1;
  if %subst(testStr: i: 1) <> %subst(testStr: j: 1);
    return *OFF;
  endif;
endfor;
```

- Cette boucle `for` parcourt la chaîne jusqu'à sa moitié.
- `j = len - i + 1`; Calcule l'indice du caractère à comparer depuis la fin de la chaîne.
- `%subst(testStr: i: 1)` et `%subst(testStr: j: 1)` extraient les caractères à comparer.
- `return *OFF`; Retourne `*OFF` (faux) si les caractères ne correspondent pas.

```
rpg
Copy code

return *ON;
end-proc;
```

- `return *ON`; Retourne `*ON` (vrai) si la chaîne est un palindrome.
- `end-proc`; Marque la fin de la procédure.

La première ligne `**FREE` n'est pas présente ici, mais dans un environnement réel, elle devrait être la première ligne du programme pour indiquer qu'il s'agit d'un code RPG Full Free.



A decorative graphic on the left side of the slide. It features several blue, 3D-style cubes of varying sizes and orientations. At the bottom, there is a grey, funnel-like structure with a blue square base. The text 'IBM i' is positioned to the right of the top-left cube.

**IBM i**

### 3. Assistants d'IA générative dans les outils de développement (Code Assistant)

# IA générative et IBM i – Les "Code Assistant"

- **1. Pour Eclipse / RDi**
  - Intégration de ChatGPT
  - Assistants d'IA au codage
  
- **2. Pour Visual Studio**
  - Intégration de ChatGPT
  - Assistants d'IA au codage
  
- **3. Solution IBM**
  - Watsonx Code Assistant



# IA générative et IBM i Les "Code Assistant"

## 1. Pour Eclipse / RDi



# a. Intégration de ChatGPT dans Eclipse / RDi

- 3 plugins

### AssistAI - A ChatGPT Plugin for Eclipse IDE

AssistAI is an Eclipse IDE plugin that brings ChatGPT functionality into your development environment. This experimental plugin has been primarily tested with Eclipse 2023/X. Features Refa...

Source Code Analyzer, Tools, General Purpose Tools, Documentation

Last Updated on Friday, September 15, 2023 - 04:47 by Wojciech Gradkowski

☆ 9    14

Install

Version alpha

---

### Remain Ai Chat ChatGPT

Introducing Remain Ai Chat by Remain Software, the ultimate AI chatbot that simplifies coding and documentation tasks in Eclipse. Its intuitive Ai Chat View and ChatGPT interface provide a seamless...

Tools, Application Development Frameworks

Last Updated on Friday, July 14, 2023 - 10:15 by Wim Jongman

☆ 4    0

Install

Version de production

---

### DBeaver - ChatGPT

ChatGPT integration for DBeaver Community (<https://marketplace.eclipse.org/content/dbeaver/>). It uses OpenAI API for smart query completion. It converts human language into SQL queries. For e...

Database Development

Last Updated on Friday, July 14, 2023 - 10:15 by Serge Rider

☆ 6    0

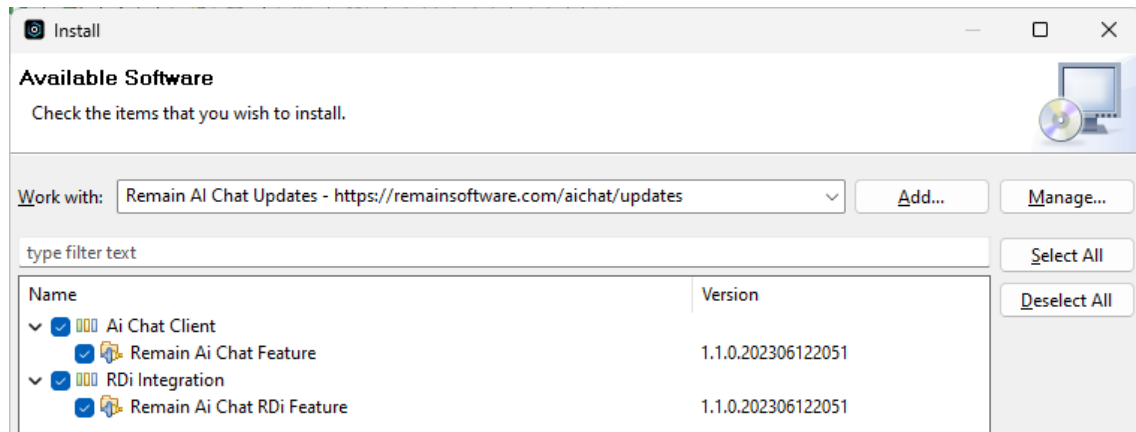
Install

Version bêta

# AI Chat for Eclipse and RDi (Remain Software)

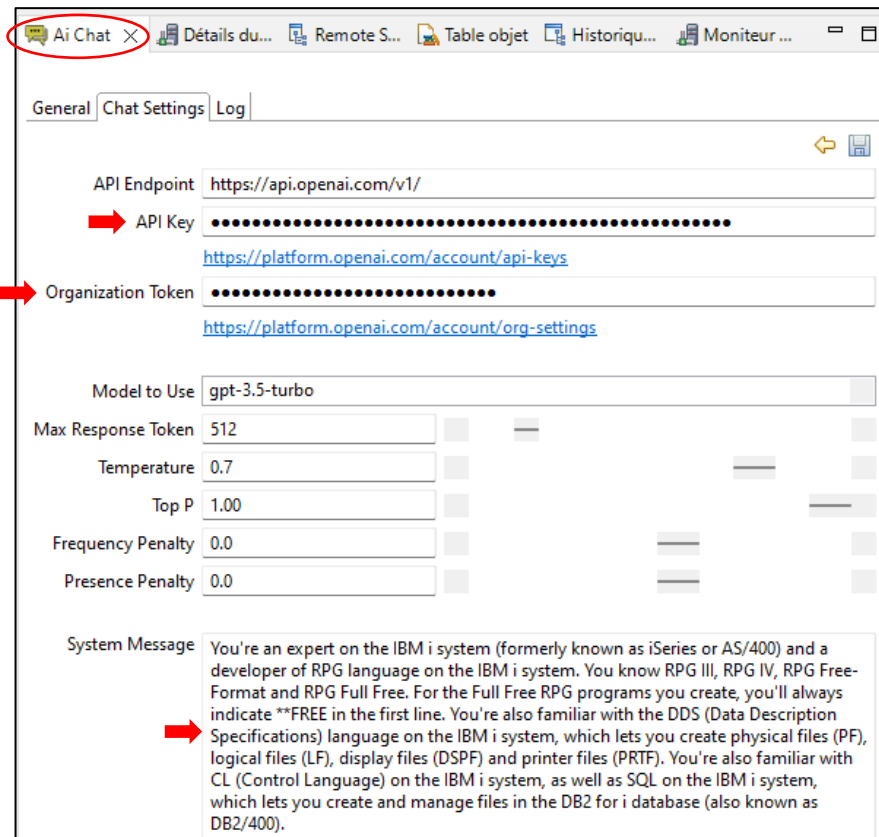
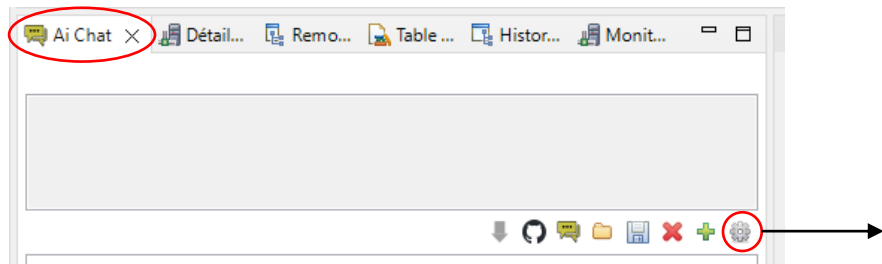
- ChatGPT intégré dans RDi
- Site Web : [https://remainsoftware.com/ai\\_chat\\_integration\\_for\\_rdi\\_and\\_eclipse](https://remainsoftware.com/ai_chat_integration_for_rdi_and_eclipse)
- Prérequis
  - RDi 9.6 ou 9.8
  - Une API Key OpenAI

## 1. Installation



# AI Chat for Eclipse and RDi (Remain Software)

## 2. Configuration



# AI Chat for Eclipse and RDi (Remain Software)

## 3. Utilisation

Can you generate the SQL source for the creation of a DB2 for i table with the following characteristics:

- Table name: FILMS
- Format name (record): FFILMS
- Column names and attributes :
  - CODE type INTEGER, type IDENTITY
  - TITRE type VARCHAR(50)
  - ANNEE NUMERIC(4)
  - CODE will be the primary key

Certainly! Here is the SQL source code to create the FILMS table with the specified characteristics:

```
...  
CREATE TABLE FILMS (  
  CODE INTEGER GENERATED ALWAYS AS IDENTITY (START WITH 1 INCREMENT BY 1),  
  TITRE VARCHAR(50),  
  ANNEE NUMERIC(4),  
  PRIMARY KEY (CODE)  
);  
...
```

This SQL code creates a table named FILMS with three columns: CODE, TITRE, and ANNEE. The CODE column is defined as an INTEGER column with the IDENTITY attribute, which means it will automatically generate unique values starting from 1. The TITRE column is defined as a VARCHAR(50) column, capable of storing up to 50 characters. The ANNEE column is defined as a NUMERIC(4) column, capable of storing a 4-digit year. The primary key constraint is set on the CODE column.

441 Tokens

# AI Chat for Eclipse and RDi (Remain Software)

## 3. Utilisation

```
// Lire le prochain enregistrement
read(e) PALINS;
enddo;

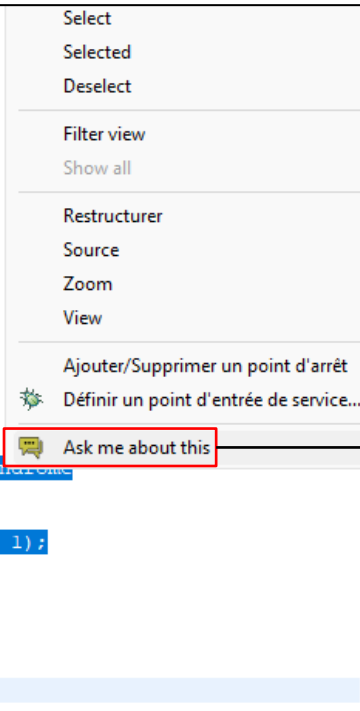
*INLR = *ON;

// Définition de la procédure
dcl-proc IsPalindrome;
  dcl-pi *n ind;
  testStr VARCHAR(50) const;
end-pi;
dcl-s i INT(5);
dcl-s j INT(5);
dcl-s len INT(5);

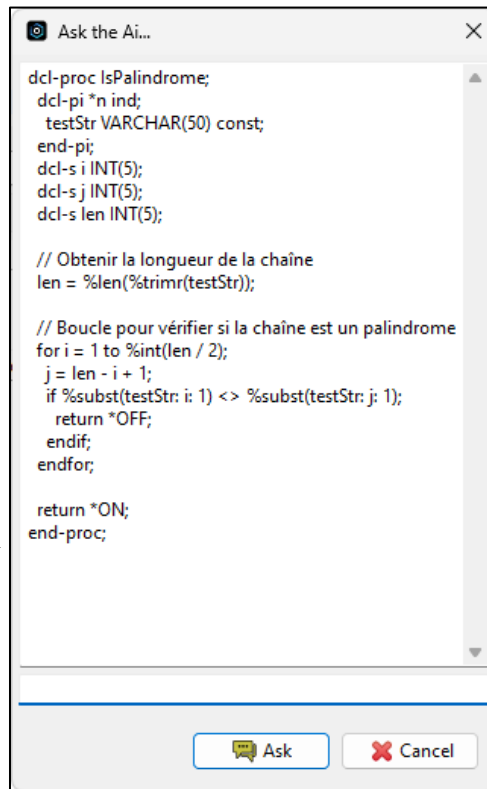
// Obtenir la longueur de la chaîne
len = %len(%trimr(testStr));

// Boucle pour vérifier si la chaîne est un palindrome
for i = 1 to %int(len / 2);
  j = len - i + 1;
  if %subst(testStr: i: 1) <> %subst(testStr: j: 1);
    return *OFF;
  endif;
endfor;

return *ON;
end-proc;
```



A context menu is shown over the code, with the option "Ask me about this" highlighted in a red box. Other options include "Select", "Selected", "Deselect", "Filter view", "Show all", "Restructurer", "Source", "Zoom", "View", "Ajouter/Supprimer un point d'arrêt", and "Définir un point d'entrée de service...".



The AI Chat dialog box, titled "Ask the Ai...", displays the code from the previous image. At the bottom, there are "Ask" and "Cancel" buttons.

```
Ask the Ai...

dcl-proc IsPalindrome;
dcl-pi *n ind;
  testStr VARCHAR(50) const;
end-pi;
dcl-s i INT(5);
dcl-s j INT(5);
dcl-s len INT(5);

// Obtenir la longueur de la chaîne
len = %len(%trimr(testStr));

// Boucle pour vérifier si la chaîne est un palindrome
for i = 1 to %int(len / 2);
  j = len - i + 1;
  if %subst(testStr: i: 1) <> %subst(testStr: j: 1);
    return *OFF;
  endif;
endfor;

return *ON;
end-proc;
```



# AI Chat for Eclipse and RDi (Remain Software)

## ■ Les +

- S'intègre avec RDi
- 2 fonctionnalités : "AI Chat" et "Ask me about this"

## ■ Les –

- Nécessite une API Key OpenAI
- ChatGPT 3.5-Turbo uniquement
- Prompt en anglais uniquement ?
  - Erreur 400 si demande en français
- Assez lent, déconnexions fréquentes (read time out)
- Pas de fonctions de complétion de code, de documentation, d'optimisation, de débogage...

## b. Assistants d'IA au codage pour Eclipse

- 2 assistants
  - aiXcoder
  - Tabnine
  
- Pas de support des langages "traditionnels" IBM i (RPG, CL, COBOL, DDS)
  
- Support des langages Java, Python...



☆ 5    💬 3

 Install

**aiXcoder Code Completer & Code Search Engine**

AiXcoder Code Completer & Code Search Engine for Java AiXcoder is a powerful code completer & code search engine based on state-of-the-art deep learning technology. It has the potential ...

Source Code Analyzer, General Purpose Tools, Other, Editor, IDE

Last Updated on Tuesday, March 14, 2023 - 03:27 by Yiyang Hao



☆ 21    💬 14

 Install

**Tabnine - AI assistant for software developers**

AI assistant for software developers Code faster with AI code completions What's Tabnine Tabnine is an AI code assistant that makes you a better developer. Tabnine will increase your...

Editor, Tools, Programming Languages

Last Updated on Wednesday, August 16, 2023 - 16:12 by Avichay Libeskind Mulyan



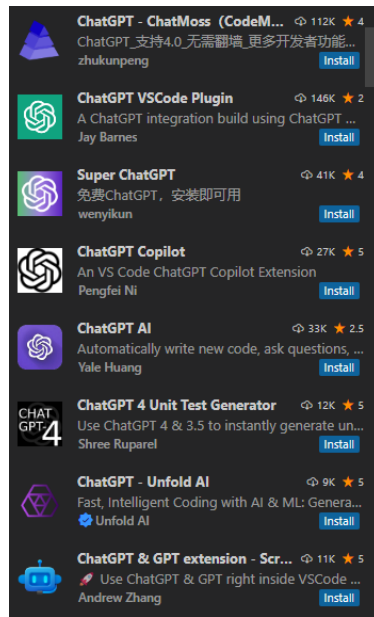
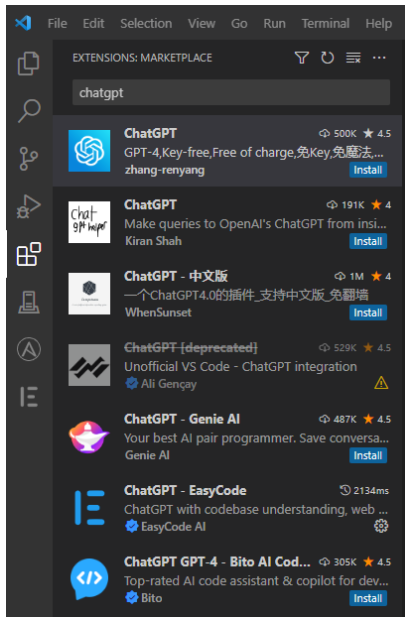
# IA générative et IBM i Les "Code Assistant"

## 2. Pour VS Code

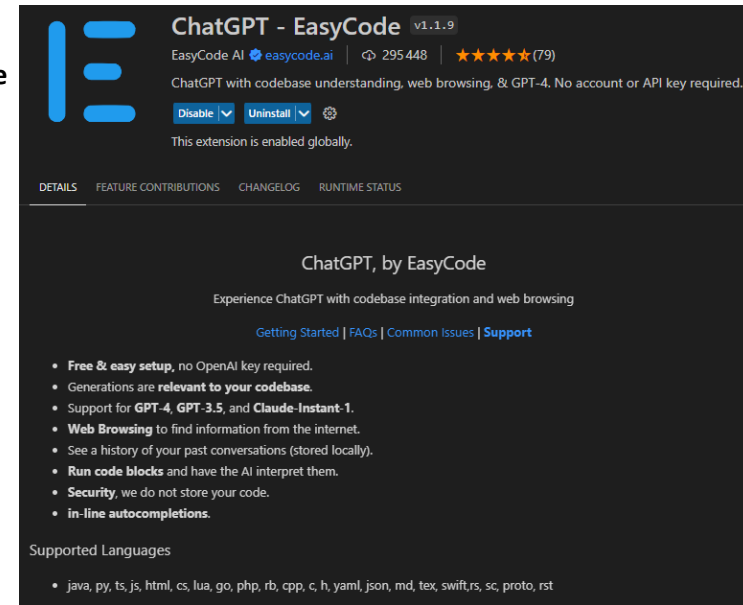


# a. Intégration de ChatGPT dans VS Code

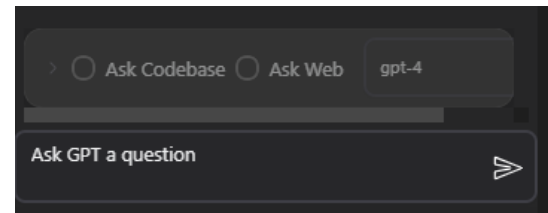
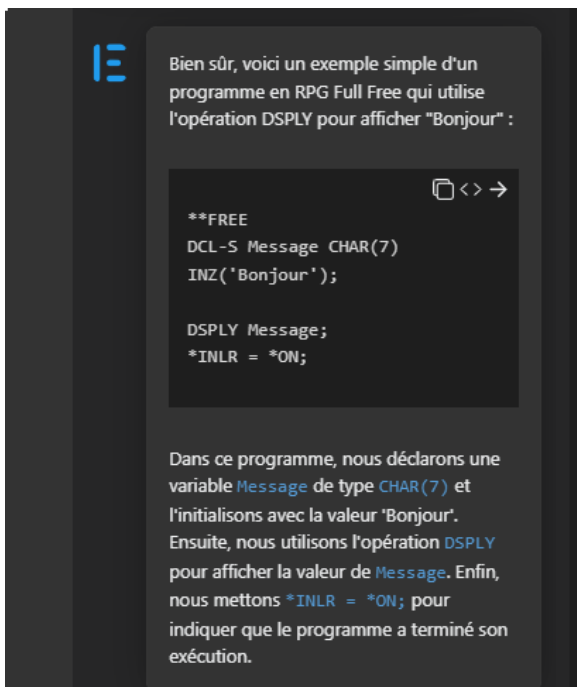
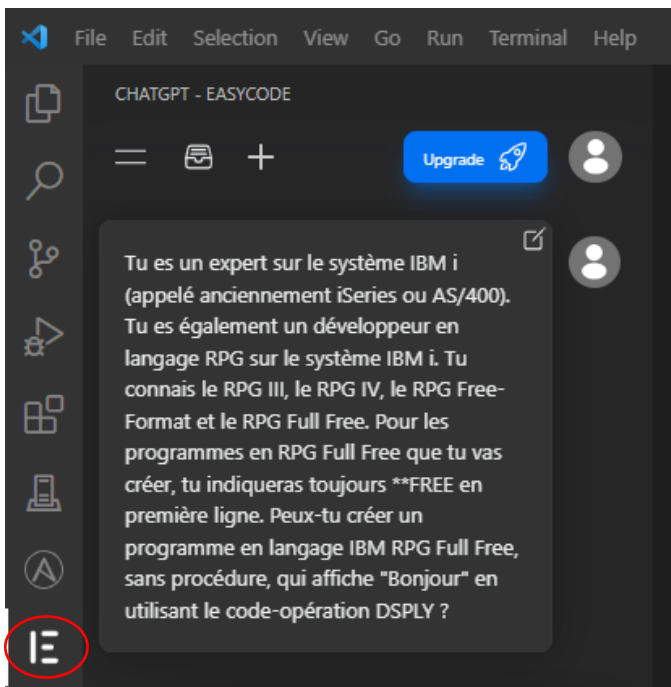
- Des dizaines d'extensions disponibles (faire une recherche sur "Chat GPT ou "Code GPT")
- Pour certains d'entre eux : support **partiel** des langages traditionnels IBM i (RPG, COBOL, CL, DDS)
  - Pas de suggestions (complétion)** dans les éditeurs, mais des **assistants** dans les menus contextuels



Exemple :  
ChatGPT – Easy Code



# Exemple avec Chat GPT – Easy Code



# Exemple avec Chat GPT – Easy Code

```
// Définition de la procédure
dcl-proc IsPalindrome;
  dcl-pi *n ind;
  testStr VARCHAR(50) const;
end-pi;
dcl-s i INT(5);
dcl-s j INT(5);
dcl-s len INT(5);

// Obtenir la longueur de la chaîne
len = %len(%trimr(testStr));

// Boucle pour vérifier si la chaîne est un palindrome
for i = 1 to %int(len / 2);
  j = len - i + 1;
  if %subst(testStr: i: 1) <> %subst(testStr: j: 1);
    return *OFF;
  endif;
endfor;

return *ON;
end-proc;
```

- Ask GPT
  - Ask GPT: Fix Bug
  - Ask GPT: What does this do?
  - Ask GPT: Refactor Code
  - Ask GPT: Write Unit Tests
  - GPT: Index Codebase
  - GPT: Clear Cache
- Command Palette... Ctrl+Shift+P

```
CHATGPT - EASYCODE
Upgrade

dcl-proc IsPalindrome;
dcl-pi *n ind;
testStr VARCHAR(50) const;
end-pi;
dcl-s i INT(5);
dcl-s j INT(5);
dcl-s len INT(5);

// Obtenir la longueur de la chaîne
len = %len(%trimr(testStr));

// Boucle pour vérifier si la chaîne est un
palindrome
for i = 1 to %int(len / 2);
j = len - i + 1;
if %subst(testStr: i: 1) <> %subst(testStr: j:
1);
return *OFF;
endif;
endfor;

return *ON;
end-proc;

Explique ce que fait ce code
```

# Exemple avec Chat GPT – Easy Code



Ce code est écrit en RPGLE, un langage de programmation utilisé sur les systèmes IBM i. Il définit une procédure nommée "IsPalindrome" qui vérifie si une chaîne de caractères donnée est un palindrome ou non.

Un palindrome est un mot, une phrase, un nombre ou une autre séquence de caractères qui peut être lu de la même manière dans les deux sens, en ignorant les espaces, la ponctuation et la capitalisation.

Voici comment fonctionne cette procédure :

Elle prend une chaîne de caractères en entrée (testStr).

Elle obtient la longueur de cette chaîne (après avoir supprimé les espaces de fin).

Elle parcourt la moitié de la chaîne en utilisant une boucle for. Pour chaque caractère, elle compare le caractère à la position actuelle avec le caractère à la position correspondante à partir de la fin de la chaîne.

Si elle trouve une paire de caractères qui ne correspondent pas, elle retourne \*OFF (qui est l'équivalent de False en RPGLE), indiquant que la chaîne n'est pas un palindrome.

Si elle parvient à parcourir toute la moitié de la chaîne sans trouver de paires de caractères non correspondantes, elle retourne \*ON (qui est l'équivalent de True en RPGLE), indiquant que la chaîne est un palindrome.

# Exemple avec Chat GPT – Easy Code

```
**FREE
ctl-opt actgrp(*new) decedit('0,');

dsply %char(proc1( 1 : 100 : 0 : 3 ));

return ;

dcl-proc proc1 ;
  dcl-pi *n zoned(15:4) ;
  z1 zoned(15:2) const ;
  z2 zoned(5:2) const ;
  z3 zoned(9:4) const ;
  z4 zoned(9:4) const ;
end-pi ;

//return %float(z1)/(%float(((z2/100)*z3)+z4));
return (z1)/(((z2/100)*z3)+z4);
end-proc ;
```

- Ask GPT
- Ask GPT: Fix Bug
- Ask GPT: What does this do?
- Ask GPT: Refactor Code
- Ask GPT: Write Unit Tests
- GPT: Index Codebase
- GPT: Clear Cache
- Command Palette... Ctrl+Shift+P

```
CHATGPT - EASYCODE
Upgrade
**FREE
ctl-opt actgrp(*new) decedit('0,');

dsply %char(proc1( 1 : 100 : 0 : 3 ));

return ;

dcl-proc proc1 ;
dcl-pi *n zoned(15:4) ;
z1 zoned(15:2) const ;
z2 zoned(5:2) const ;
z3 zoned(9:4) const ;
z4 zoned(9:4) const ;
end-pi ;

//return
%float(z1)/(%float(((z2/100)*z3)+z4));
return (z1)/(((z2/100)*z3)+z4);
end-proc ;

Ce programme ne fonctionne pas. Le code-
opération DSPLY affiche 0,0000 alors qu'il
devrait afficher 0,3333. Peux-tu le corriger ?
```



# Exemple avec Chat GPT – Easy Code



Le problème est que vous effectuez une division entière, ce qui signifie que le résultat est arrondi à l'entier le plus proche. Pour obtenir un résultat décimal, vous devez convertir au moins un des opérandes en float avant de faire la division. Voici comment vous pouvez corriger votre code :

```

**FREE
ctl-opt actgrp(*new) decedit('0,');

dsply %char(proc1( 1 : 100 : 0 : 3 )) ;

return ;

dcl-proc proc1 ;
  dcl-pi *n zoned(15:4) ;
    z1 zoned(15:2) const ;
    z2 zoned(5:2) const ;
    z3 zoned(9:4) const ;
    z4 zoned(9:4) const ;
  end-pi ;

  return %float(z1)/(%float(((z2/100)*z3)+z4));
end-proc ;

```

Dans ce code, j'ai ajouté la fonction `%float` pour convertir `z1` et l'expression `((z2/100)*z3)+z4` en float avant de faire la division. Cela devrait résoudre votre problème et afficher le résultat attendu de 0,3333.

## b. Assistants d'IA au codage pour VS Code

- Bito (Bito)
- CodeWhisperer (Amazon)
- **Copilot (GitHub et OpenAI)**
- Replit (Replit)
- Tabnine (Tabnine)
- ...
  
- Je ne les ai pas tous testés, mais a priori **aucun** ne propose actuellement le **support complet** des langages **RPG, COBOL, CL, DDS**
  - J'ai testé **Copilot** qui propose un support partiel

# Exemple avec GitHub Copilot

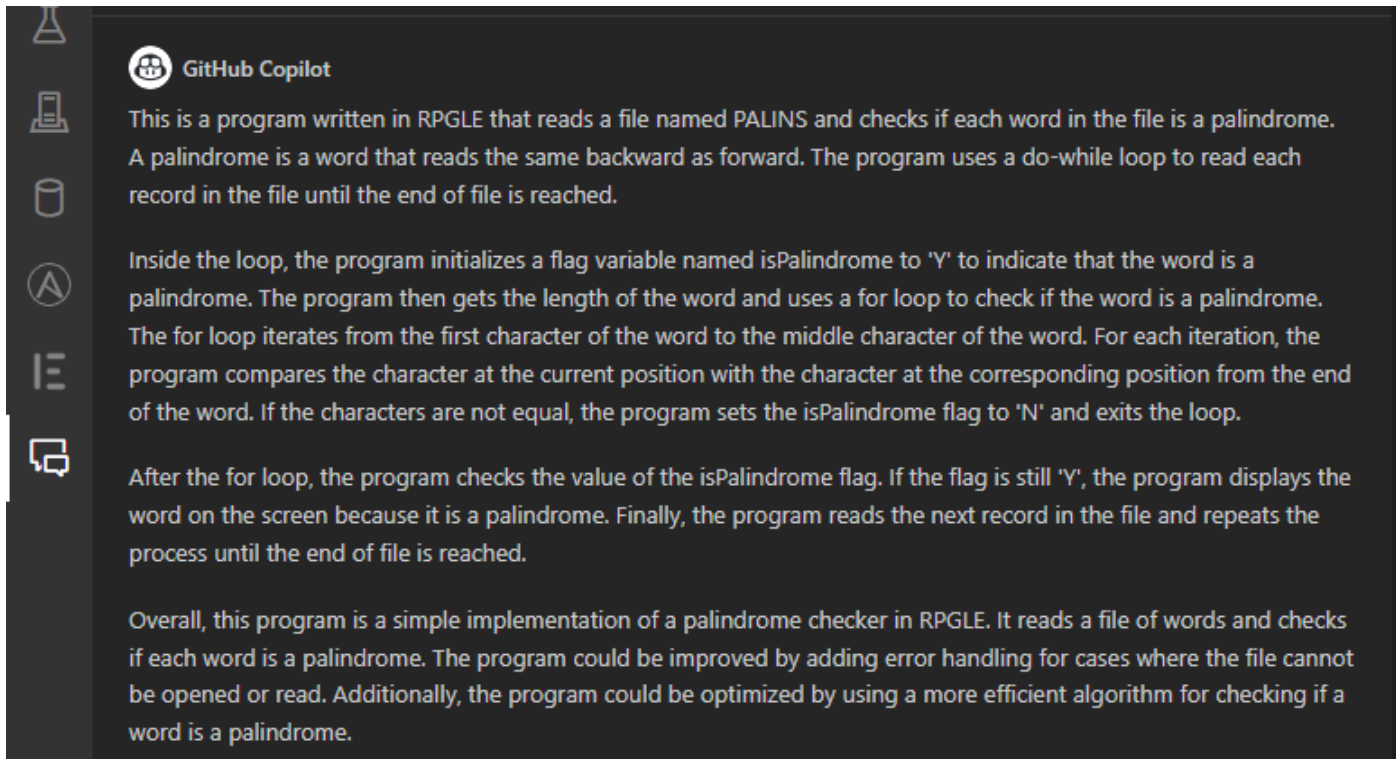
```


11 // Lire le fichier PALINS
12 read(e) PALINS;
13 dow not %eof;
14
15 // Réinitialiser l
16 isPalindrome = 'Y'
17
18 // Obtenir la long
19 len = %len(%trimr(
20
21 // Boucle pour vér
22 for i = 1 to %int(
23     j = len - i + 1;
24     if %subst(MOT: i
25         isPalindrome =
26         leave;
27     endif;
28 endfor;
29
30 // Afficher le mot

```

Atteindre la définition	F12	
Atteindre les implémentations	Ctrl+F12	
Atteindre les références	Maj+F12	
Aperçu	>	
Rechercher toutes les références	Maj+Alt+F12	
Rechercher toutes les implémentations		
<b>Copilot</b>	>	Démarrer la conversation de code Ctrl+I
Modifier toutes les occurrences	Ctrl+F2	<b>Expliquer ceci</b>
Mettre le document en forme	Maj+Alt+F	Corriger ceci
Remanier...	Ctrl+Maj+R	Générer les documents
Convert to Free Format		Générer des tests

# Exemple avec GitHub Copilot



 GitHub Copilot

This is a program written in RPGLE that reads a file named PALINS and checks if each word in the file is a palindrome. A palindrome is a word that reads the same backward as forward. The program uses a do-while loop to read each record in the file until the end of file is reached.

Inside the loop, the program initializes a flag variable named isPalindrome to 'Y' to indicate that the word is a palindrome. The program then gets the length of the word and uses a for loop to check if the word is a palindrome. The for loop iterates from the first character of the word to the middle character of the word. For each iteration, the program compares the character at the current position with the character at the corresponding position from the end of the word. If the characters are not equal, the program sets the isPalindrome flag to 'N' and exits the loop.

After the for loop, the program checks the value of the isPalindrome flag. If the flag is still 'Y', the program displays the word on the screen because it is a palindrome. Finally, the program reads the next record in the file and repeats the process until the end of file is reached.

Overall, this program is a simple implementation of a palindrome checker in RPGLE. It reads a file of words and checks if each word is a palindrome. The program could be improved by adding error handling for cases where the file cannot be opened or read. Additionally, the program could be optimized by using a more efficient algorithm for checking if a word is a palindrome.

# Exemple avec GitHub Copilot

```
7
8 dcl-proc proc1 ;
9   dcl-pi *n zoned(15:4) ;
10   z1 zoned(15:2) const ;
11   z2 zoned(5:2) const ;
12   z3 zoned(9:4) const ;
13   z4 zoned(9:4) const ;
14   end-pi ;
15
16 //return %float(z1)/%fl
17   return (z1)/(((z2/100)
18 end-proc ;
19
20
```

Atteindre la définition	F12
Atteindre les implémentations	Ctrl+F12
Atteindre les références	Maj+F12
Aperçu	>
Rechercher toutes les références	Maj+Alt+F12
Rechercher toutes les implémentations	
<b>Copilot</b>	>
Modifier toutes les occurrences	Ctrl+F2
Mettre le document en forme	Maj+Alt+F
Remanier...	Ctrl+Maj+R
Convert to Free Format	

Démarrer la conversation de code
Expliquer ceci
<b>Corriger ceci</b>
Générer les documents
Générer des tests

```
   return (z1)/(((z2/100)*z3)+z4);
end-proc ;
```

/fix

Accepter Abandonner

Unrecognized language: rpgle

# Exemple avec GitHub Copilot

☰ COPILOT1.RPGLE ✕

CHATGPT\_PB > QRPGLSRC > ☰ COPILOT1.RPGLE

1 Appuyez sur `Ctrl` + `I` pour demander à GitHub Copilot Chat de faire quelque chose. Commencez à taper pour ignorer.

1

Déclare une procédure nommée CALCUL



Copilot generated code may be incorrect

☰ COPILOT1.RPGLE ●

CHATGPT\_PB > QRPGLSRC > ☰ COPILOT1.RPGLE

```
1  ∨ dcl-proc CALCUL;  
2      // Code de la procédure CALCUL  
3  end-proc;  
4
```

Déclare une procédure nommée CALCUL



Accepter

Abandonner



Lignes 4 modifiées





# IBM watsonx Code Assistant



# L'offre IBM watsonx



- IBM **watsonx** : plateforme d'**IA** pour les entreprises
- Trois composants :
  - **watsonx.ai**
    - Studio (gestion des modèles, IA générative, Machine Learning)
  - **watsonx.data**
    - Magasin de données pour l'IA
  - **watsonx.governance**
    - Kits d'outils pour créer des flux d'IA
- Solutions IBM qui s'appuient sur l'IA générative de **watsonx** :
  - **Watsonx Orchestrate** (automatisation des tâches et des processus des entreprises)
  - **Watsonx Assistant** (création de chatbots)
  - **Watsonx Code Assistant** (génération de code, conversion de code...)



# IBM watsonx Code Assistant

Reinventing how work gets done | Application modernization solution

## Watson Code Assistant

Write code faster with AI-generated recommendations based on natural language inputs

Proven results:

**30%**

Productivity gain in application modernization

**Accelerate** code development through AI-powered natural language processing and increase developer productivity



**Enhance** the quality of code, with greater efficiency and improved accuracy



**Narrow** the IT skills gap by accelerating developer onboarding, making software development more accessible



**Reduce** time to value for automation and improve development cycles with AI code recommendations



**Maintain** high levels of accuracy and transparency through data-source attribution



**Tune** the **watsonx** foundation model with your own data set to apply organizational best practices



# IBM watsonx Code Assistant

- Site Web : <https://www.ibm.com/products/watsonx-code-assistant>
  
- Annonce du 26/10/2023, deux fonctionnalités :
  - **IBM watsonx Code Assistant for Ansible Lightspeed**
    - Assistant pour la génération et la compréhension de playbooks Ansible
  - **IBM watsonx Code Assistant for Z**
    - Cartographie applicative, références croisées, dépendances entre composants existants
    - Restructuration (refactoring) du code COBOL, extraction des services métier
    - Conversion des services COBOL en code Java optimisé pour z/OS, CICS, IMS et DB2
    - Génération automatique des tests unitaires
  
- Dans le futur ?
  - **IBM watsonx Code Assistant for i ?**



**IBM i**



A vous maintenant

# Qu'en pensez-vous ?

- Trouvez-vous un intérêt à l'IA générative pour le développement sous IBM i ?
- Seriez-vous intéressé par :
  - Un assistant d'IA pour RDi ?
  - Un assistant d'IA pour VS Code for i ?
  - Une version "Watsonx Code Assistant **for i**" ?

## Pour terminer

PB

Peux-tu écrire un sonnet sur l'IBM i dans le style de Shakespeare



Dans le donjon de bits, de silicium noir,  
Réside IBM i, machine sans égards,  
Avec RPG, sa lance, DDS, son bouclier,  
Il déjoue les pièges de l'ennui, à la journée.

CL est son scribe, SQL son conseiller,  
Pour gérer DB2, ils sont sans pareil.  
En Free-Format ils chantent, en Full Free ils dansent,  
Un ballet de codes, un langage d'espérance.

Mais écoute ! Qu'est-ce que j'entends là, mon ami ?  
L'IBM i murmure, tel un ancien esprit.  
"Il est temps, braves humains, de comprendre mon sort,  
Je suis le passé et l'avenir, le début et le port."

Oui, IBM i, ton règne n'est pas terminé,  
Dans nos cœurs de code, tu resteras à jamais.

