Tutorial 1 Mise en place du lab.

Table des matières

- Objectifs
- Ressources
- Pré-Requis

Objectifs

Ce TP utilise de nombreuses technologies (xml,http,rpg,sql,rest,...).

Afin de faciliter les manipulations nous allons utiliser Visual studio Code.

VSC (visual studio code) est un IDE universel et open source. Il est relativement rapide, peu gourmand et gratuit.

Mais ce n'est pas son seul atout, il est extensible.

Ainsi il peut facilement s'adapter aux besoins des utilisateurs (développeur, testeur, rédacteur de doc,.....) avec l'installation d'extensions disponible sur une **market place**

Pour ce faire nous avons de réaliser quelques manipulations et installations d'outils.

Ainsi nous allons:

- Installer visual studio code sur votre poste client.
- Installer les différentes extensions nécessaires (code for IBMi) via un profile gist ?
- Ajouter un .profile et .bashrc
- Configurer git (?)
- Cloner le dépôt [M13_P12] en local.
- Configurer notre connexion
- Configurer notre projet
- Compiler notre premier programme
- Lancer notre premier script sql

Ressources

- Environnement
- Temps: 45 mn.

Pré-Requis

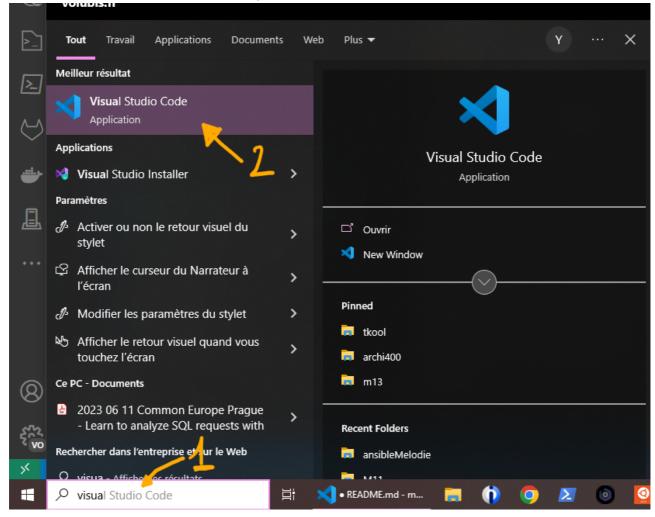
· Accéder à internet.

Énoncé

Etape 1 Installer el lancer Visual Studio Code sur le poste client.



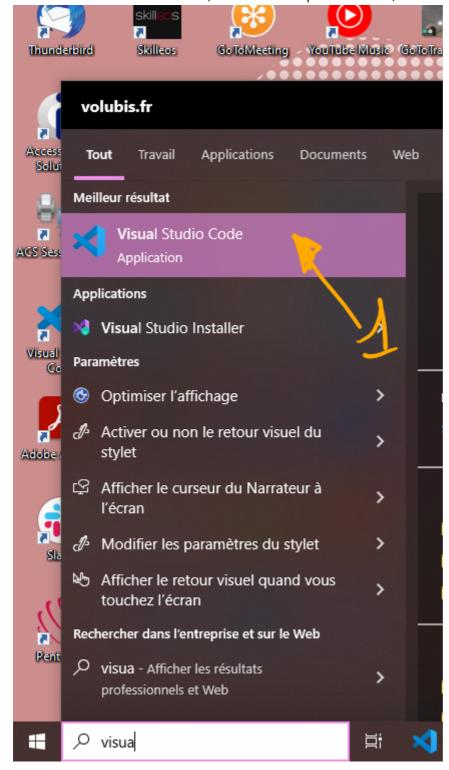
1. Rechercher Visual studio code sur votre poste client.

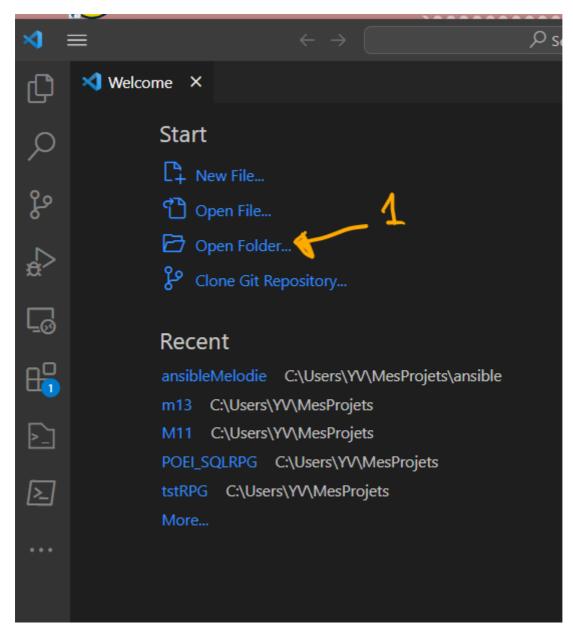


si :

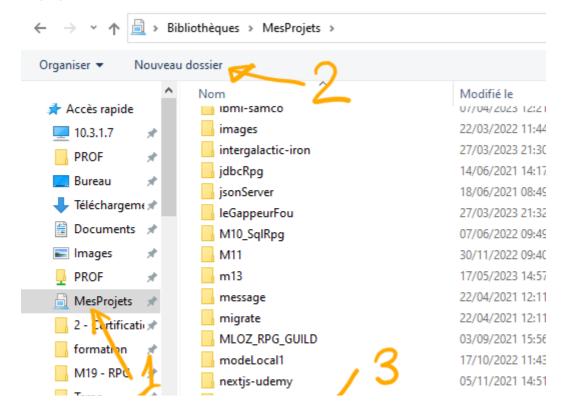
- il existe passer en 4 pour le lancer.
- o sinon passer en 2 pour l'installer.
- 2. Download Visual Studio Code
- 3. Lancer l'installation.
 - (a) lors de l'installation, choisissez l'option pour avoir vsc en clic droit et le raccourci sur le bureau.

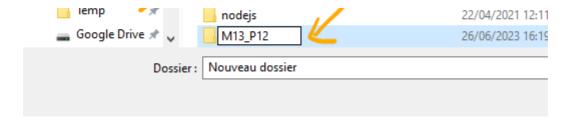
4. Ouvrer VSC et créer un dossier (ou vous voulez pour travailler)

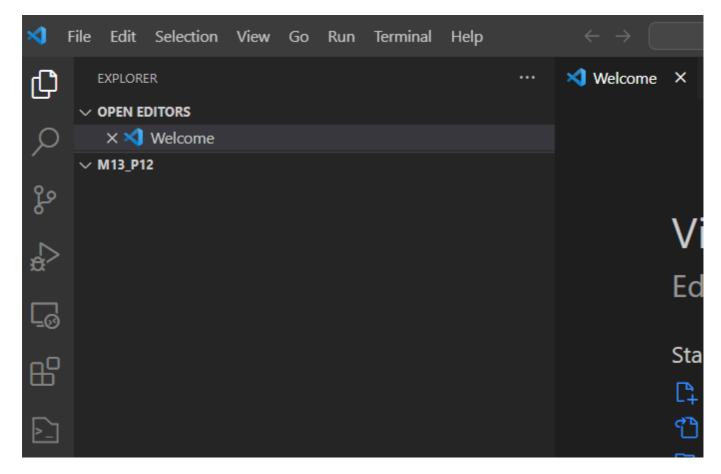






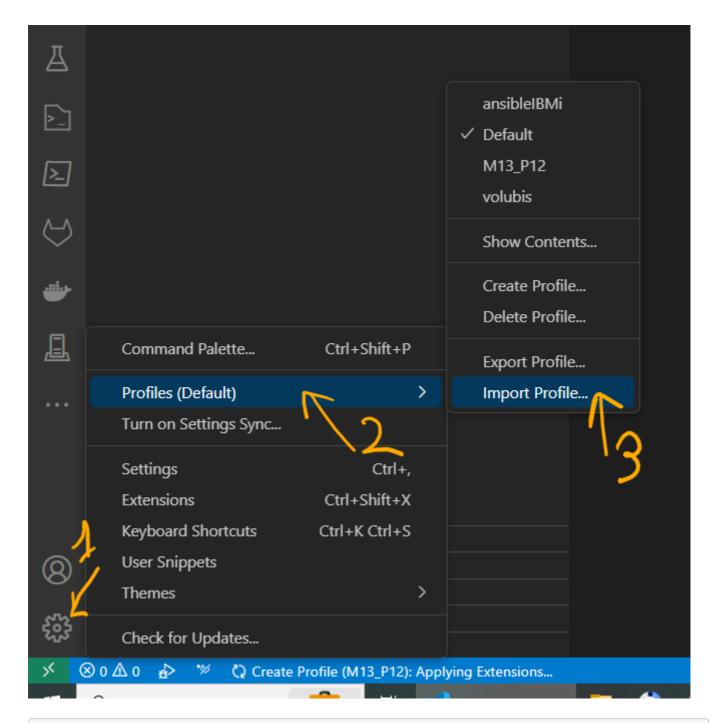






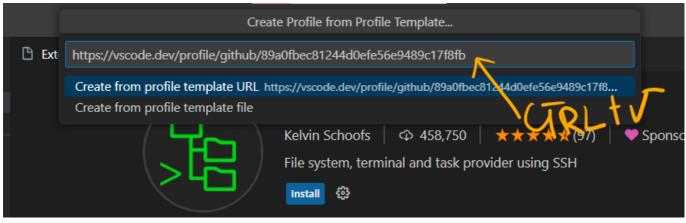
Etape 2 Installer code for IBMi.

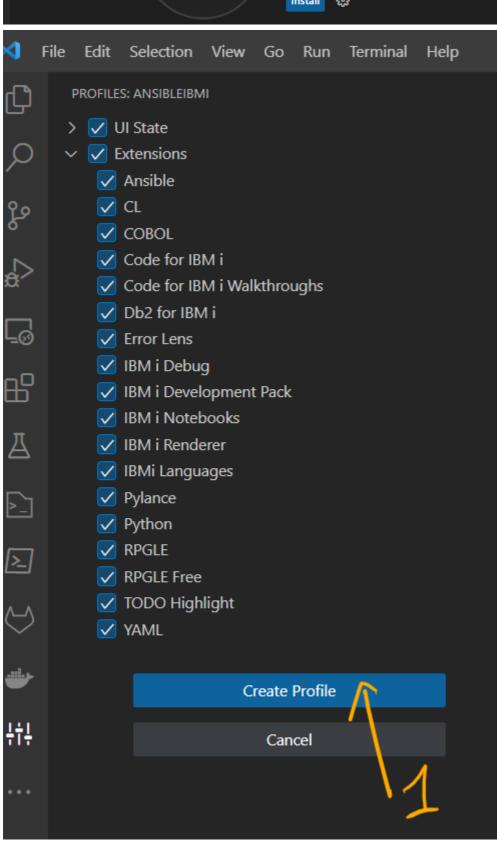
1.installation des extensions via un profile. profile Ansible

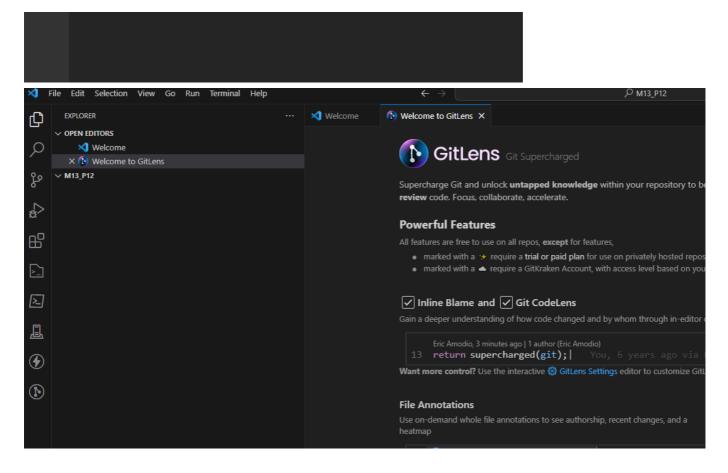


CTRL-C

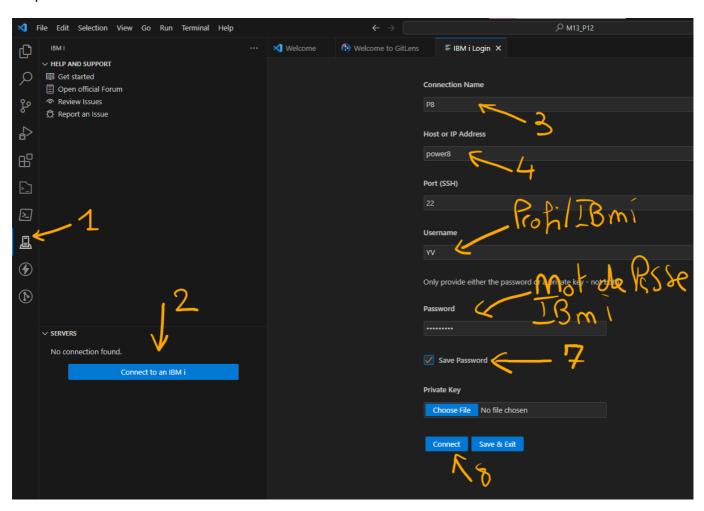
https://vscode.dev/profile/github/c06f3ba02b988ca7822e6b2aca0ac37f



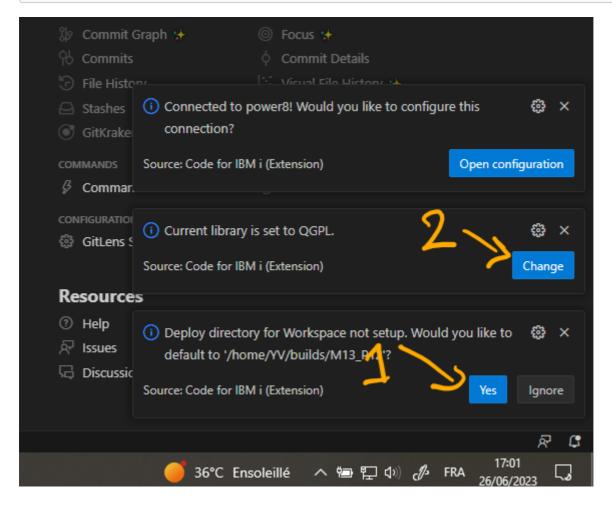


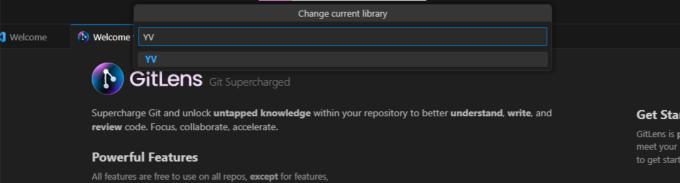


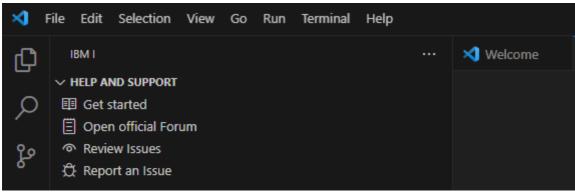
Etape 3 Se connecter à l'IBMi.

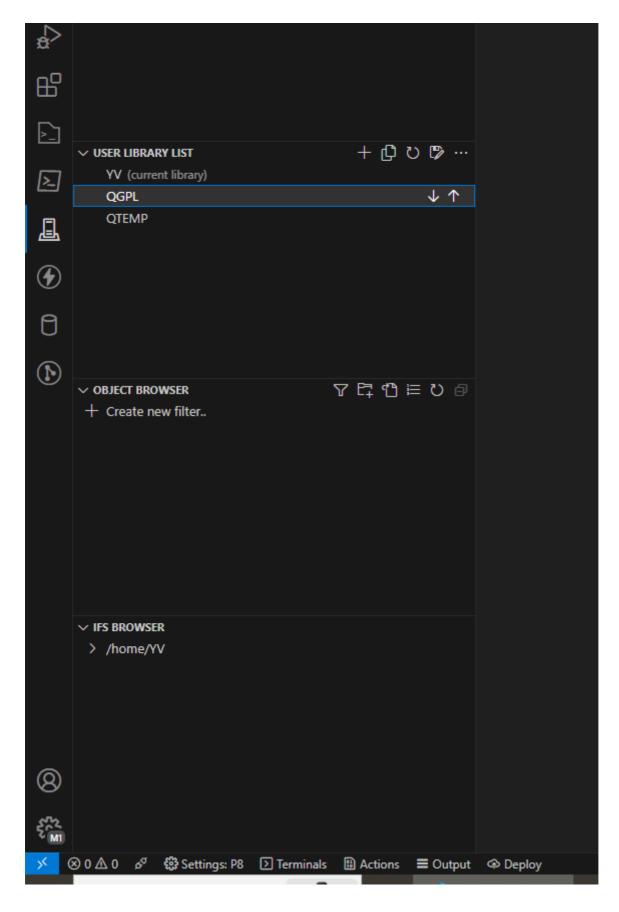


connection name : `ceQueVousVoulez`
host ou IP: `HOST_NAME_IBMI`
userName : `VotreProfilIBMi`
password : `votrePasswordIBMi`
savePassword : ok







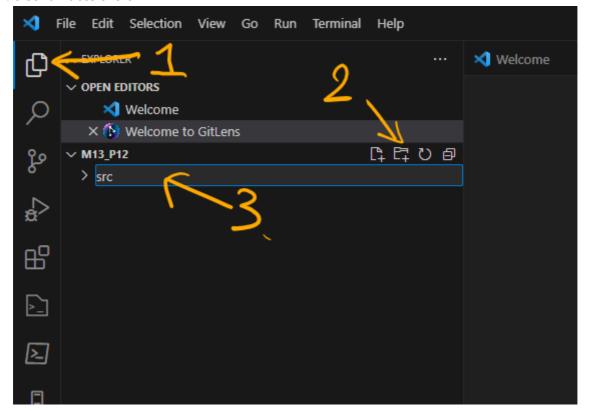


Etape 4 configurer votre projet.

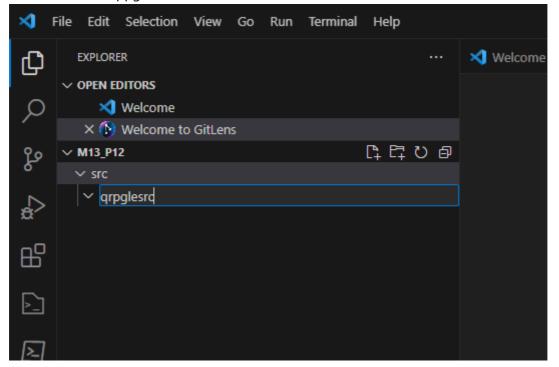
- 1. création d'un filtre pour vos sources.
- 2. ajout d'un profil.
- 3. modification liste de bibliothèques *LIBL

Etape 5 Hello World.

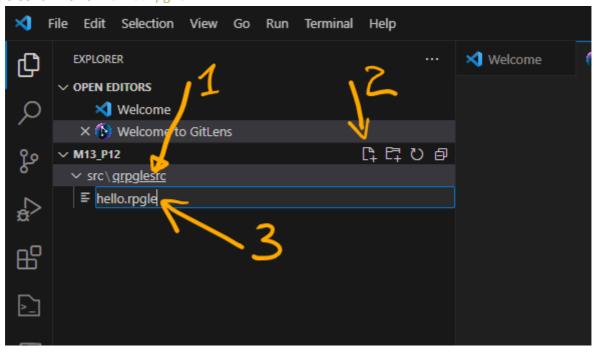
1. créer un dossier src



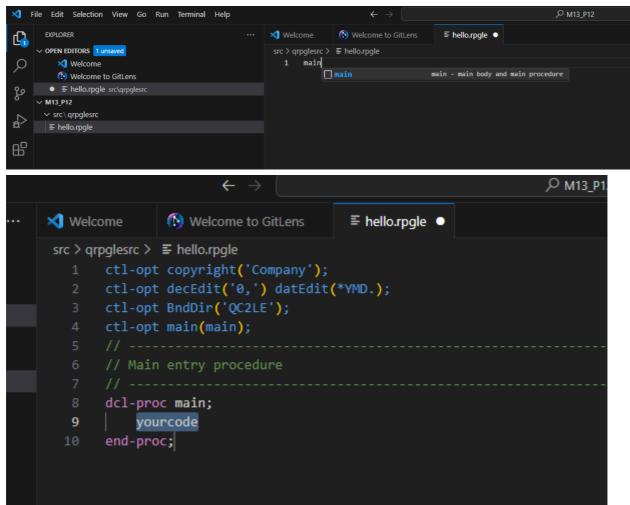
2. créer un dossier qrpglesrc dans src



3. créer un fichier hello.rpgle



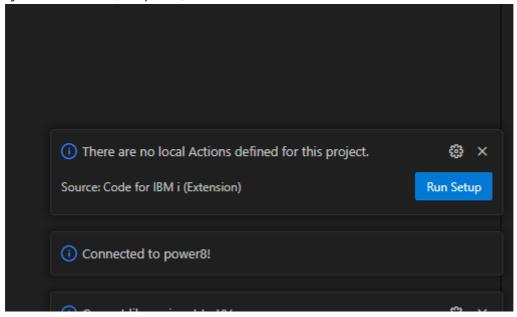
- 4. écrire **free sur la premiere colonne de la première ligne.
- 5. écrire main sur la seconde ligne du fichier et taper CTRL-SPACE et la laisser faire la magie de code for IBMi



ajouter la ligne de code

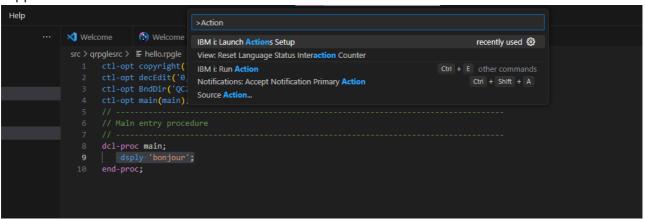
dsply 'bonjour'; sauvegarder via la touche CTRL-S

6. ajout des actions (compile,....)

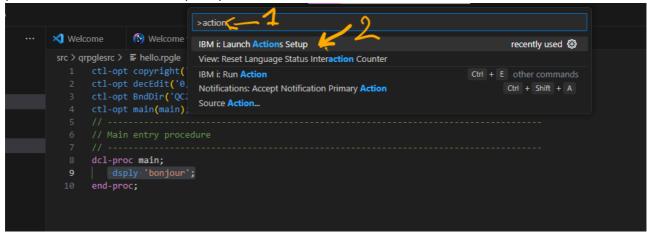


ou

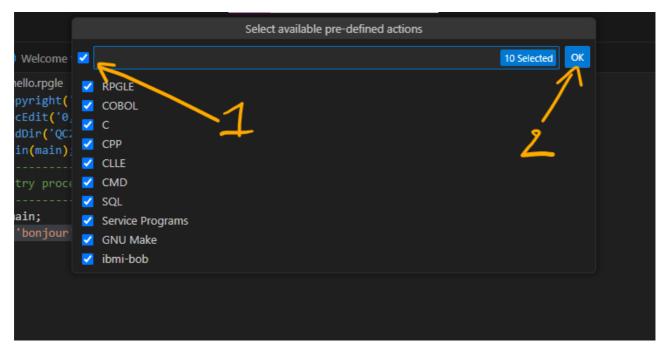
. appeler le menu des commandes de vsc via la touche F1



puis . inscrivez action sur le prompt



. sélectionnez toutes les options



sauvegarder via la touche CTRL-S et fermer le fichier

```
Welcome

Welcome

Welcome to GitLens

Fello.rpgle

vscode > {} actions.json > ...

"name": "Create RPGLE Program",
 "command": "CRTBNDRPG PGM(&CURLIB/&NAME) SRCSTMF('&RELATIVEPATH') OPTION(*EVENTF) Description

deployFirst": true,
 "environment": "ile",
 "extensions": [
 "RPGLE"

]

| "name": "Create RPGLE Module",
 "command": "CRTRPGMOD MODULE(&CURLIB/&NAME) SRCSTMF('&RELATIVEPATH') OPTION(*EVENTF)

| "name": "Create RPGLE Module",
 "command": "CRTRPGMOD MODULE(&CURLIB/&NAME) SRCSTMF('&RELATIVEPATH') OPTION(*EVENTF)

| "command": "CRTRPGMOD MODULE(&CURLIB/&NAME) SRCSTMF('&RELATIVEPATH') OPTION(*EVENTF)
| "Tommand": "CRTRPGMOD MODULE(&CURLIB/&NAME) SRCSTMF('&RELATIVEPATH')
| "Tommand": "CRTRPGMOD MODULE(&CURLIB/&NAME) SRCSTMF('&RELATIVEPATH') OPTION(*EVENTF)
| "Tommand": "CRTRPGMOD MODULE(&CURLIB/&NAME) SRCSTMF('&RELATIVEPATH') OPTION(*EVENTF)
| "Tommand": "CRTRPGMOD MODULE(&CURLIB/&NAME) SRCSTMF('&RELATIVEPATH') OPTION(*EVENTF)
| "Tommand": "CRTRPGMOD MODULE(&CURLIB/&NAME) SRCSTMF('&RELATIVEPATH')
| "Tommand": "CRTRPGMOD MODU
```

7. lancer la compile de hello

```
Select deployment method to /home/YV/builds/M13_P12
X Welcome
                  Welcome
                                 Compare Synchronizes using MD5 hash comparison
src > qrpglesrc > 

hello.rpgle
                                 Changes 2 changes detected since last upload.
        ctl-opt copyright(
                                 Working Changes Unstaged changes in git
        ctl-opt decEdit('0
                                 Staged Changes
        ctl-opt BndDir('QC2
                                  All Every file in the local workspace
        ctl-opt main(main);
        dcl-proc main;
   9
         end-proc;
```

```
Entrée de commandes VOLUBIS

Niveau de demande: 4

Détail des commandes et messages précédents:

Travail 166766/YV/QPADEV0005 démarré le 27/06/23 à 08:54:47 dans le sous-système QINTER de QSYS; soumis le 27/06/23 à 08:54:47.

> /* */
3 > call qcmd
4 > call yv/hello
DSPLY bonjour
```

Etape 6 KO world!

modifier le programme hello pour ajouter une erreur.
 nous allons pour l'exemple utiliser une variable non définie

```
dcl-proc main;
  dsply message;
end-proc;
```

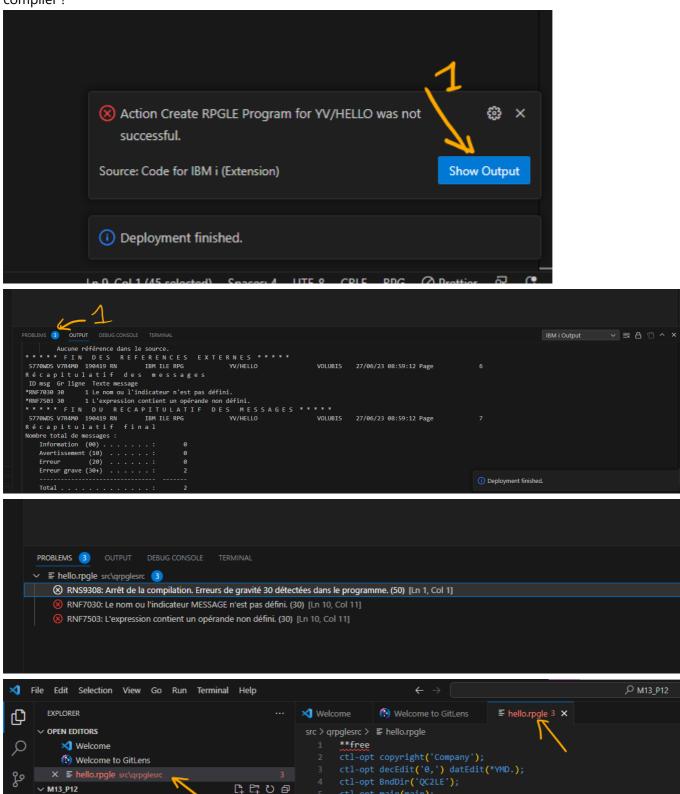
compiler!

.vscode

{} actions.json ✓ src\qrpglesrc

■ hello.rpgle

\$



ctl-opt main(main);

end-proc;

corriger et recompiler..

cool non?

Conclusion et feed-back

Correction



