**Enterprise Architect – Code – Exécution Analyzer**

Table des matières

[1 Préparer Enterprise Architect (EA13) pour l'Execution Analyzer 4](#_Toc24741182)

[1.1 Configurer les variables d'environnement (Local Paths) dans Enterprise Architect (EA13) 4](#_Toc24741183)

[1.1.1 Accéder aux variables d'anvironnement (Local Paths) dans EA13 : 4](#_Toc24741184)

[1.1.2 ajouter/modifier une variable locale (Local Path) dans EA13 : 5](#_Toc24741185)

[1.1.3 Les principales variables d'environnement (Local Paths) utiles 5](#_Toc24741186)

[1.1.3.1 JAVA\_HOME 5](#_Toc24741187)

[1.1.3.2 MAVEN\_REPOSITORY 6](#_Toc24741188)

[1.1.3.3 Le path du projet ECLIPSE pour lequel on veut installer l'Execution Analyzer 7](#_Toc24741189)

[2 Récupérer la ligne de commande de l'application sous Exécution Analyzer dans ECLIPSE 8](#_Toc24741190)

[2.1 Objectif de la récupération de la ligne de commande 8](#_Toc24741191)

[2.2 Actions à mener pour récupérer la ligne de commande dans ECLIPSE 9](#_Toc24741192)

[2.2.1 Positionner un point d'arrêt dans la méthode main() de la classe applicative 9](#_Toc24741193)

[2.2.2 Lancer l'application en mode DEBUG dans ECLIPSE 10](#_Toc24741194)

[2.2.3 Clicker sur "Properties" du Thread javaw.exe dans la perspective DEBUG d'ECLIPSE 11](#_Toc24741195)

[2.2.4 Récupérer la command line 12](#_Toc24741196)

[2.2.5 Copier la command line dans un editeur Notepad++ 13](#_Toc24741197)

[3 Paramétrer la tâche Run dans EA13 14](#_Toc24741198)

[3.1 Structure du RUN dans EA13 14](#_Toc24741199)

[3.1.1 Lanceur exécutable Java (machine virtuelle du JDK) 14](#_Toc24741200)

[3.1.2 Répertoire des classes compilées 15](#_Toc24741201)

[3.1.3 Classpath intégrant toutes les bibliothèques utilisées (jar) 16](#_Toc24741202)

[3.1.4 Une classe application (point d'entrée de l'application avec une méthode main()) 16](#_Toc24741203)

[3.1.5 Résumé de la syntaxe de la ligne de commande du RUN 17](#_Toc24741204)

[3.2 Modifier la ligne de commande originale ECLIPSE 17](#_Toc24741205)

[3.2.1 Retirer les ajouts d'ECLIPSE 17](#_Toc24741206)

[3.2.2 Substituer les variables d'environnement (Local Paths) aux chemins en dur 17](#_Toc24741207)

[3.2.2.1 Substituer %JAVA\_HOME% à C:\Program Files\Java\jdk1.8.0\_231 18](#_Toc24741208)

[3.2.2.2 Substituer éventuellement java.exe à javaw.exe 18](#_Toc24741209)

[3.2.2.3 Substituer %PATH\_PROJET\_uploading-file-springboot% à D:\Donnees\eclipse\eclipseworkspace\uploading-file-springboot 19](#_Toc24741210)

[3.2.2.4 Substituer %MAVEN\_REPOSITORY% à C:\Users\dan\.m2\repository 19](#_Toc24741211)

[3.3 Injecter la ligne de commande dans la tâche RUN de EA13 20](#_Toc24741212)

[3.4 Tester la tâche RUN depuis EA13 21](#_Toc24741213)

[4 Paramétrer la tâche DEBUG dans EA13 23](#_Toc24741214)

[4.1 Structure de DEBUG dans EA13 23](#_Toc24741215)

[4.1.1 Java RunTime Environment (JRE) du JDK 23](#_Toc24741216)

[4.1.2 Répertoire des classes compilées 24](#_Toc24741217)

[4.1.3 Répertoire des sources 25](#_Toc24741218)

[4.1.4 Classpath intégrant toutes les bibliothèques utilisées (jar) 26](#_Toc24741219)

[4.1.5 Résumé de la syntaxe de la ligne de commandes du DEBUG 26](#_Toc24741220)

[4.2 Modifier la ligne de commande original d'ECLIPSE 27](#_Toc24741221)

[4.2.1 Retirer les ajouts d'ECLIPSE 27](#_Toc24741222)

[4.2.2 Substituer les variables d'environnement (Local Paths) aux chemins en dur 27](#_Toc24741223)

[4.2.2.1 Substituer %JAVA\_HOME% à C:\Program Files\Java\jdk1.8.0\_231 28](#_Toc24741224)

[4.2.2.2 Substituer jre=%JAVA\_HOME%\jre à "%JAVA\_HOME%\bin\javaw.exe" 28](#_Toc24741225)

[4.2.2.3 Rajouter une virgule après Dfile.encoding=UTF-8 28](#_Toc24741226)

[4.2.2.4 Substituer -Djava.class.path= à -classpath 28](#_Toc24741227)

[4.2.2.5 Mettre tous le classpath (jars) entre quotes 28](#_Toc24741228)

[4.2.2.6 Substituer %PATH\_PROJET\_uploading-file-springboot% à D:\Donnees\eclipse\eclipseworkspace\uploading-file-springboot 29](#_Toc24741229)

[4.2.2.7 Substituer %MAVEN\_REPOSITORY% à C:\Users\dan\.m2\repository 29](#_Toc24741230)

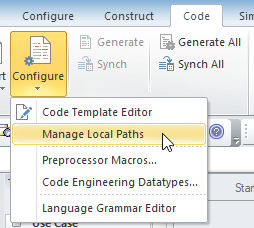
[4.2.2.8 RAJOUTER LE CHEMIN DES SOURCES %PATH\_PROJET\_uploading-file-springboot%\src\main\java dans le classpath 30](#_Toc24741231)

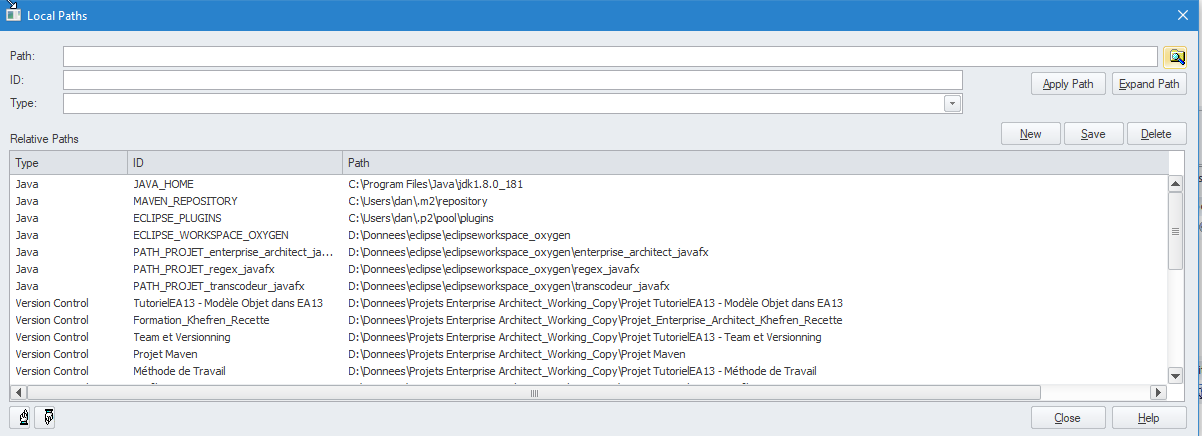
# Préparer Enterprise Architect (EA13) pour l'Execution Analyzer

## Configurer les variables d'environnement (Local Paths) dans Enterprise Architect (EA13)

### Accéder aux variables d'anvironnement (Local Paths) dans EA13 :

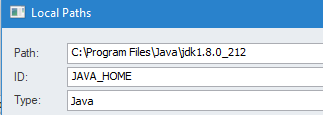
Ruban **code** > Menu **Configure** > **Manage Local Paths**



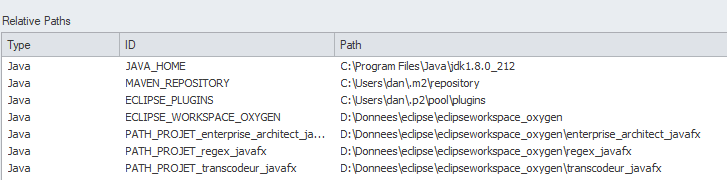


### ajouter/modifier une variable locale (Local Path) dans EA13 :

1. Sélectionner le path à l'aide du bouton explorer
2. Donner le nom voulu à la variable locale
3. Choisir le type "Java"



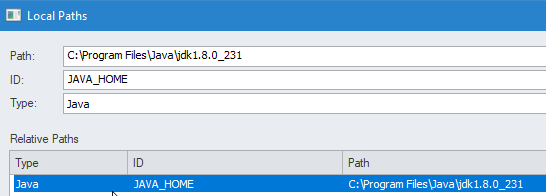
Puis **Save**.



### Les principales variables d'environnement (Local Paths) utiles

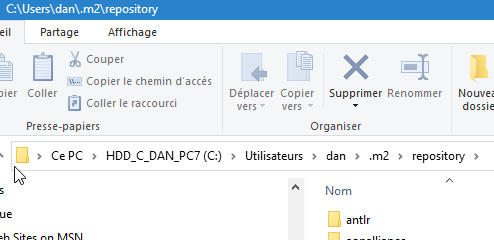
#### JAVA\_HOME

Pointe vers le **JDK** sur la machine. Est aussi une variable d'environnement Windows.

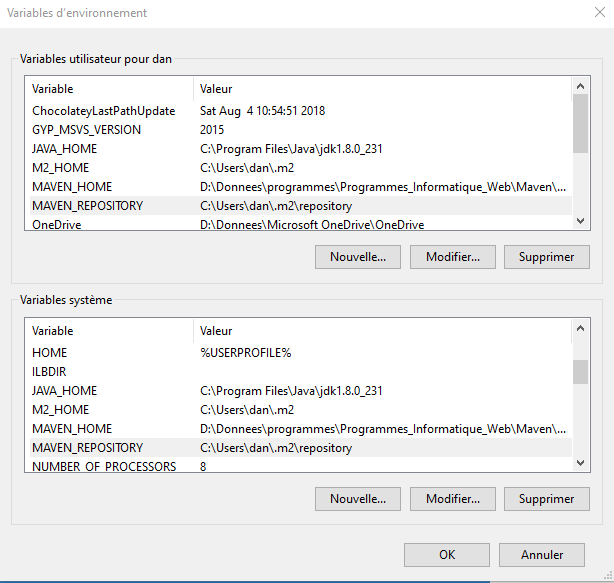


#### MAVEN\_REPOSITORY

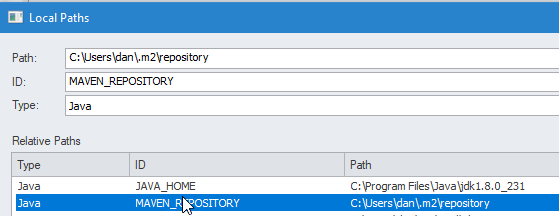
Pointe vers le **repository Maven** sur la machine.



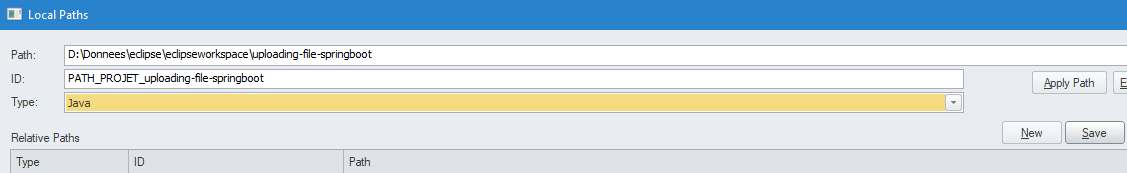
Est aussi une variable d'environnement Windows :

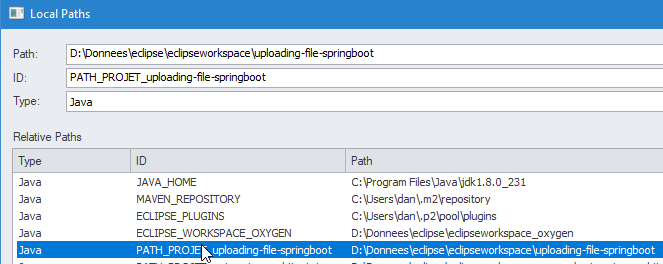


Dans EA13 :



#### Le path du projet ECLIPSE pour lequel on veut installer l'Execution Analyzer





# Récupérer la ligne de commande de l'application sous Exécution Analyzer dans ECLIPSE

## Objectif de la récupération de la ligne de commande

L'Execution Analyzer d'EA13 permet de :

* Lancer une application Java depuis EA13 (tâche RUN)
* Deboger une application Java depuis EA13 (tâche DEBUG)
* Construire une application Java créée dans EA13 dans un dossier Windows (tâche BUILD)
* Etc…

Pour ce faire, l'Execution Analyzer *utilise Java en mode ligne de commande*.

Par exemple, pour lancer une application Java (RUN), l'Execution Analyzer a besoin qu'on lui fournisse une ligne de commande du type :

*cheminJDK*/**java.exe** **-classpath** "*chemin des .class*;*librairies dans le classpath*" *package*.**MainClass**

où :

* **java.exe** (ou **javaw.exe** pour une application fenêtrée) est le **lanceur JAVA**
* **- classpath** indique le chemin des .class et de toutes les **librairies** (jar) utilisées par l'application
* **MainClass** est la classe JAVA **point d'entrée de l'application** (classe possédant une méthode main(String[] pArgs)).

Il est fastidieux de créer manuellement cette ligne de commande, et notamment de collecter toutes les librairies (jar) utilisées par l'application.

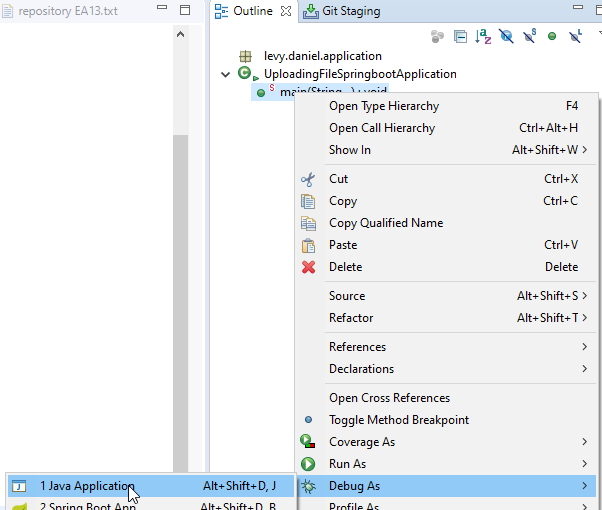
L'utilisation d'Eclipse en mode DEBUG permet de **récupérer automatiquement la ligne de commande** après avoir mis un point d'arrêt et lancé l'application en mode DEBUG.

## Actions à mener pour récupérer la ligne de commande dans ECLIPSE

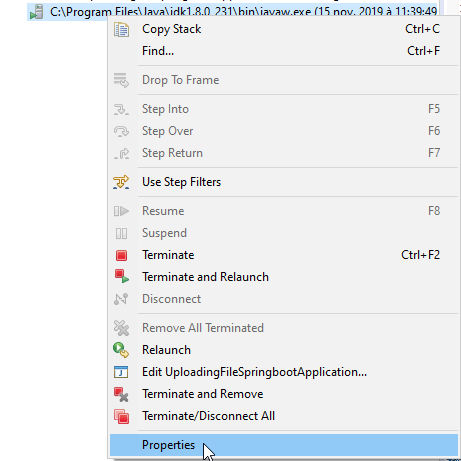
### Positionner un point d'arrêt dans la méthode main() de la classe applicative



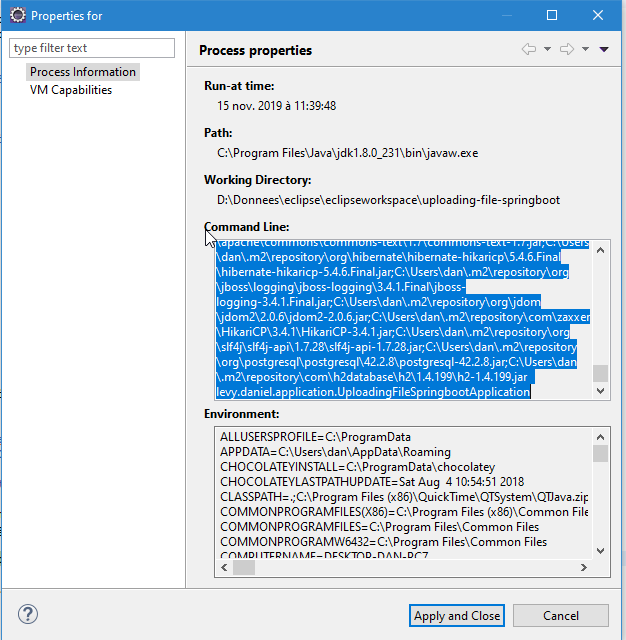
### Lancer l'application en mode DEBUG dans ECLIPSE



### Clicker sur "Properties" du Thread javaw.exe dans la perspective DEBUG d'ECLIPSE



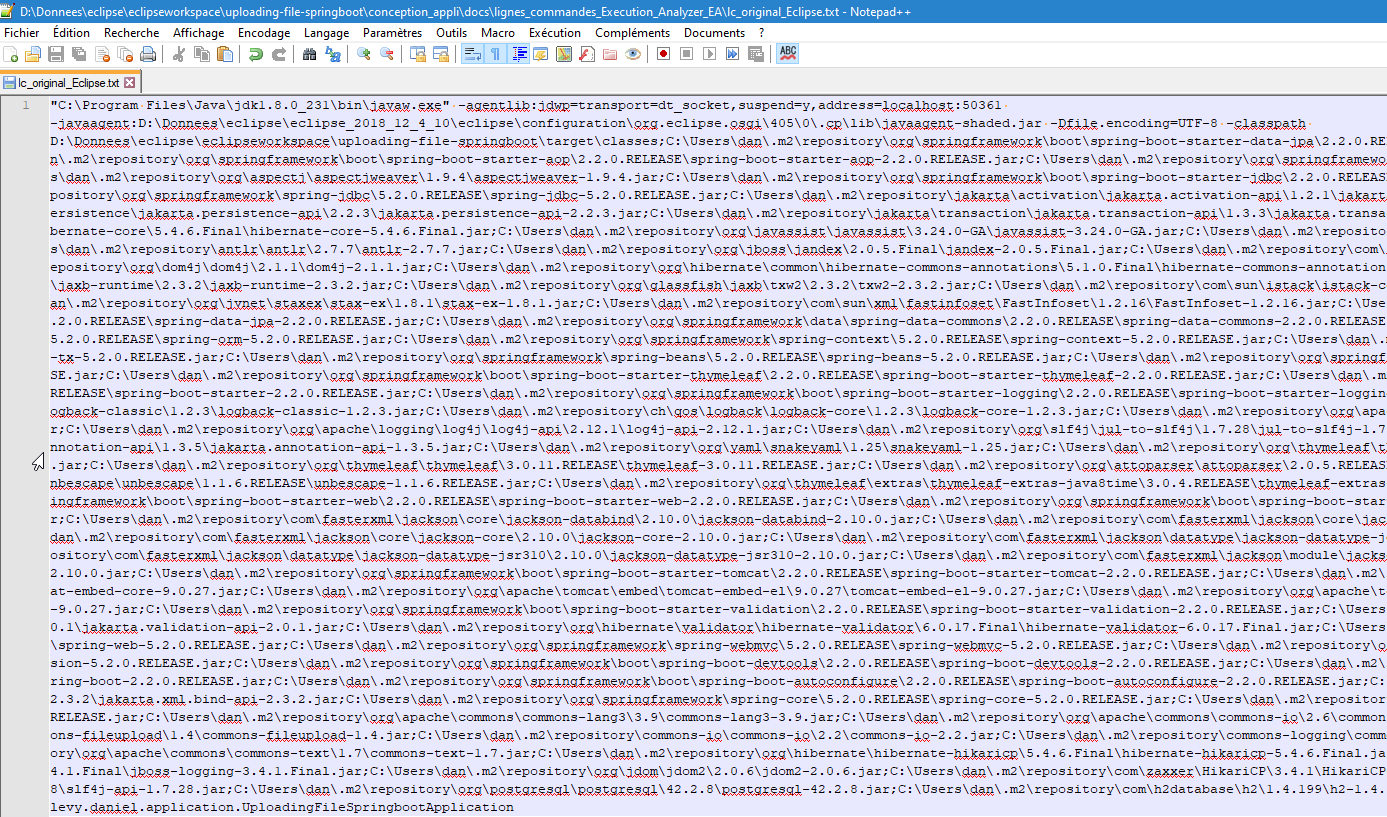
### Récupérer la command line



CTRL+A puis CTRL+C pour copier toute la ligne de commande dans le presse-papier.

Arrêter le débogage dans ECLIPSE

### Copier la command line dans un editeur Notepad++



Sauvegarder provisoirement dans Notepad++ cette ligne de commande générée par ECLIPSE.

Elle servira de point de départ pour fabriquer les lignes de commandes à fournir à EA13 pour les tâches RUN, DEBUG, …

Recopier intégralement cette ligne dans 2 nouveaux éditeurs Notepad++ pour préparer le RUN et le DEBUG



# Paramétrer la tâche Run dans EA13

## Structure du RUN dans EA13

EA13 a besoin pour lancer une application Java de :

### Lanceur exécutable Java (machine virtuelle du JDK)

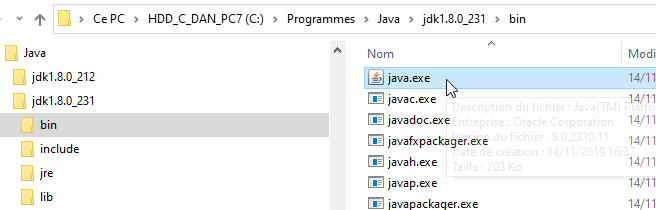
* java.exe pour une application non graphique (application non fenêtrée),
* javaw.exe pour une application graphique (AWT, Swing, JavaFX, …).

S'agissant dans l'exemple d'une application Java non graphique Spring Boot, la ligne de commande du RUN doit commencer par :

"%JAVA\_HOME%\bin\java.exe"

*Le chemin de l'exécutable Java est mis entre parenthèse au cas où il y aurait des espaces dans le chemin* (comme dans 'Program Files').

Ce qui correspond au bon chemin sur le disque dur puisque %JAVA\_HOME% sera traduit dans EA13 par : C:\Program Files\Java\jdk1.8.0\_231 grâce aux Local Paths. On le vérifie dans l'explorateur Windows.



### Répertoire des classes compilées

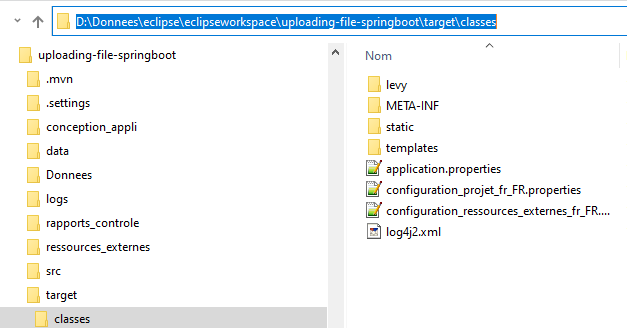
EA13 devant lancer un projet Java, il faut évidemment qu'il ait **accès au code Java compilé** (les .class).

S'agissant d'un projet MAVEN dans l'exemple, ces classes compilées sont buildées par MAVEN dans un répertoire :

%PATH\_PROJET\_uploading-file-springboot%\target\classes

Pour mémoire, %PATH\_PROJET\_uploading-file-springboot% sera (normalement !) traduit dans EA13 par D:\Donnees\eclipse\eclipseworkspace\uploading-file-springboot grâce aux Local Paths.

On peut vérifier que ce chemin pointe bien sur les classes compilées sur le disque dur dans l'explorateur Windows :



**Ce répertoire des classes compilées doit être intégré dans le classpath de la ligne de commande.**

### Classpath intégrant toutes les bibliothèques utilisées (jar)

Un classpath se déclare dans une ligne de commande RUN Java avec -classpath suivi du chemin des classes compilées et de tous les chemins de toutes les bibliothèques (jar) le tout entre double-quotes ".

La ligne de commande devient donc à ce stade :

"%JAVA\_HOME%\bin\javaw.exe" -classpath %PATH\_PROJET\_uploading-file-springboot%\target\classes;*tous les chemins de tous les jar*"

On y trouve bien :

1. l'exécutable java javaw.exe,
2. l'ordre -classpath ,
3. Puis dans le classpath : la référence aux classes compilées
4. Et, toujours dans le classpath, la référence à tous les jars.

* **Le classpath est mis entre double-quotes "** au cas où il y aurait des espaces dans le chemin des jar.
* **Les chemins des jar sont séparés par des point-virgule** ";".

On utilise le fichier dans Notepad "ligne\_commande\_run.txt" contenant la ligne de commande générée par Eclipse pour récupérer tous les chemins de tous les jars.

### Une classe application (point d'entrée de l'application avec une méthode main())

Puisque le RUN dans le script EA13 doit lancer l'application, il doit connaître le chemin de la **classe applicative** qui possède une méthode main(String[] pArgs).

On doit rajouter dans la ligne de commande le *chemin Java (avec des séparateurs "." et pas "/")* de la classe applicative par rapport à la racine des sources Java (src ou src/main/java si projet Java MAVEN). Cette classe applicative doit être indiquée *après le classpath*.

* La classe doit être indiquée **sans extension .class**.
* Le chemin Java commence à la racine du projet compilé (bin ou target/classes). C'est donc le chemin à partir de src pour un projet Java simple ou src/main/java pour un projet MAVEN.

Dans l'exemple :

levy.daniel.application.UploadingFileSpringbootApplication

### Résumé de la syntaxe de la ligne de commande du RUN

"[Lanceur java]" -classpath "[chemin des build];[tous chemins de tous les jar]" [classe applicative]

## Modifier la ligne de commande originale ECLIPSE

**Objectif** : modifier la ligne de commande récupérée dans ECLIPSE pour pouvoir piloter l'application Java depuis EA13.

### Retirer les ajouts d'ECLIPSE

Eclipse rajoute pour son débogage l'instruction :

-agentlib:jdwp=transport=dt\_socket,suspend=y,address=localhost:63044 …

Il faut donc retirer ces instructions dans l'éditeur RUN de Notepad++

Le début de la ligne de commande originale :

"C:\Program Files\Java\jdk1.8.0\_231\bin\javaw.exe" -agentlib:jdwp=transport=dt\_socket,suspend=y,address=localhost:50361 -javaagent:D:\Donnees\eclipse\eclipse\_2018\_12\_4\_10\eclipse\configuration\org.eclipse.osgi\405\0\.cp\lib\javaagent-shaded.jar -Dfile.encoding=UTF-8 -classpath

Devient donc :

"C:\Program Files\Java\jdk1.8.0\_231\bin\javaw.exe" -Dfile.encoding=UTF-8 -classpath

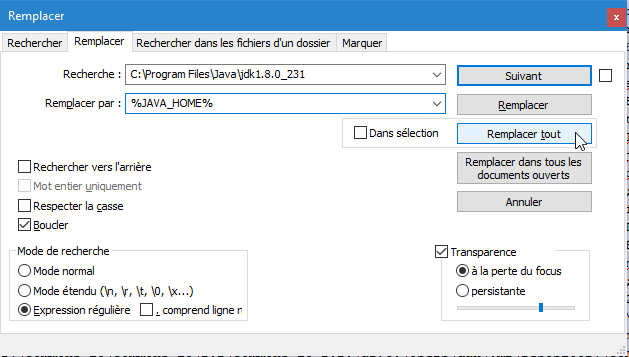
### Substituer les variables d'environnement (Local Paths) aux chemins en dur

**Objectif** : Cette étape optionnelle a l'intérêt de simplifier la ligne de commande résultante et d'améliorer la maintenabilité de la ligne de commande sur différentes machines. L'utilisation des variables d'environnement permet en effet de s'affranchir de la localisation réelle sur chaque machine du JDK, du dépôt MAVEN, …

Les variables d'environnement **s'utilisent toujours comme en MS-DOS entourées de %**.

On substitue donc aux chemins en dur leurs variables d'environnement dans l'éditeur RUN de Notepad ++.

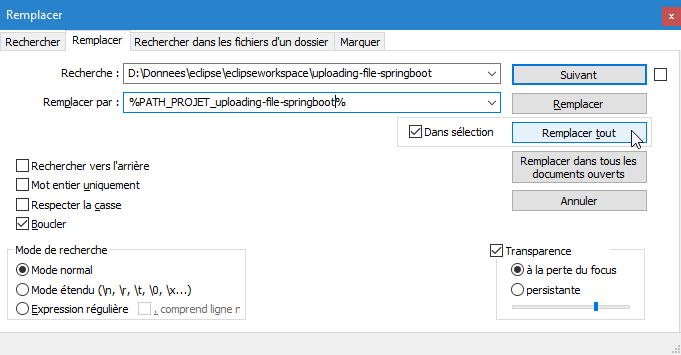
#### Substituer %JAVA\_HOME% à C:\Program Files\Java\jdk1.8.0\_231



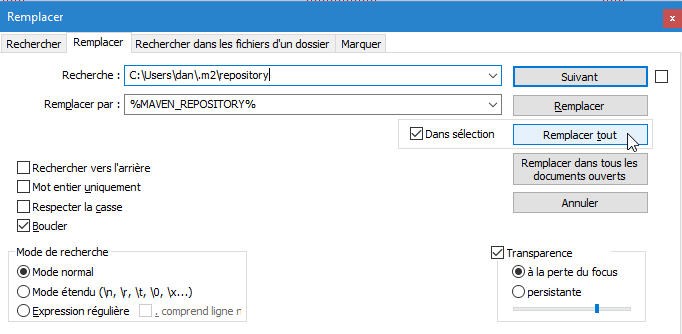
#### Substituer éventuellement java.exe à javaw.exe

Si l'application n'est pas fenêtrée (application JavaFX ou Swing), le lanceur doit être java.exe à la place de javaw.exe.

#### Substituer %PATH\_PROJET\_uploading-file-springboot% à D:\Donnees\eclipse\eclipseworkspace\uploading-file-springboot

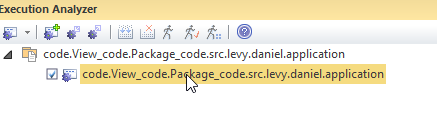


#### Substituer %MAVEN\_REPOSITORY% à C:\Users\dan\.m2\repository

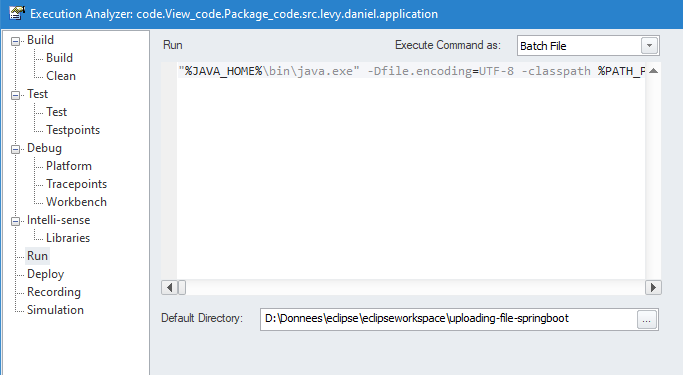


## Injecter la ligne de commande dans la tâche RUN de EA13

Ruban **Code** > Menu **Analyzer** > **Analyzer Scripts > double clicker sur le script concerné**



1. Ouvrir l'onglet Run dans la fenêtre Exécution Analyzer du script concerné.
2. Copier-Coller la ligne de commande modifiée dans NotePad++ dans la zone de texte sur une seule ligne
3. Indiquer le chemin vers le projet ECLIPSE dans la zone "Default Directory"

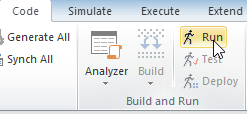


Puis OK pour sauvegarder ce script RUN.

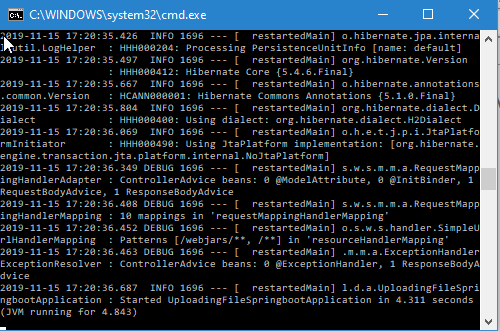
## Tester la tâche RUN depuis EA13

**Objectif** : vérifier que le script RUN précédemment enregistré permet de piloter l'application depuis EA13

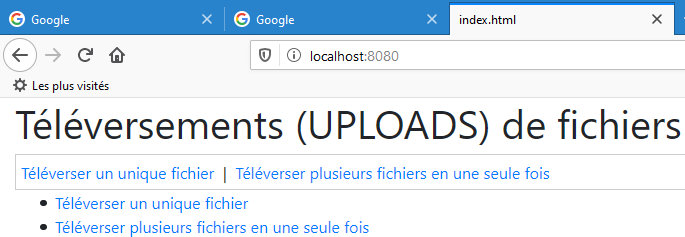
Menu **Code** > **Run**



La fenêtre de commande Windows s'ouvre montrant que l'application s'est bien lancée :



S'agissant ici d'une application Web SPRING BOOT, on vérifie dans un navigateur que l'application répond bien :



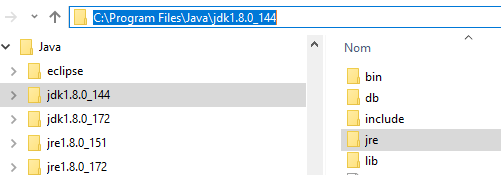
# Paramétrer la tâche DEBUG dans EA13

## Structure de DEBUG dans EA13

EA13 a besoin pour lancer une application Java en mode DEBUG de :

### Java RunTime Environment (JRE) du JDK

Pour Debugger, il faut le **JRE du JDK** (pas le simple JRE) :



On peut utiliser optionnellement une variable d'environnement WINDOWS et/ou un Local Path :

La ligne de commande du DEBUG doit commencer par :

jre=%path\_jdk%\jre **Pas de double quotes " autour de cette instruction**

puis un séparateur virgule ","

puis (opt.) l'instruction concernant l'encodage de la plateforme : -Dfile.encoding=UTF-8

puis un séparateur virgule ","

A ce stade, le début de la ligne de commandes de DEBUG est :

jre=%path\_jdk%\jre, -Dfile.encoding=UTF-8,

### Répertoire des classes compilées

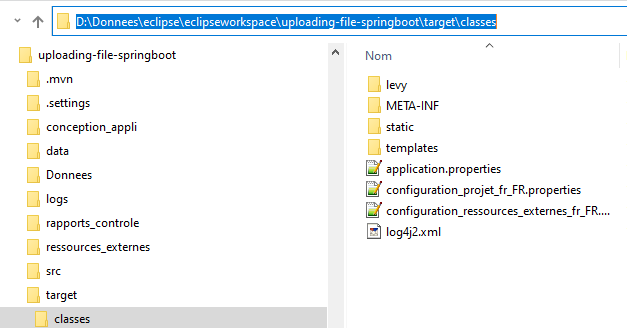
EA13 devant lancer un projet Java, il faut évidemment qu'il ait **accès au code Java compilé** (les .class).

S'agissant d'un projet MAVEN dans l'exemple, ces classes compilées sont buildées par MAVEN dans un répertoire :

%PATH\_PROJET\_uploading-file-springboot%\target\classes

Pour mémoire, %PATH\_PROJET\_uploading-file-springboot% sera (normalement !) traduit dans EA13 par D:\Donnees\eclipse\eclipseworkspace\uploading-file-springboot grâce aux Local Paths.

On peut vérifier que ce chemin pointe bien sur les classes compilées sur le disque dur dans l'explorateur Windows :



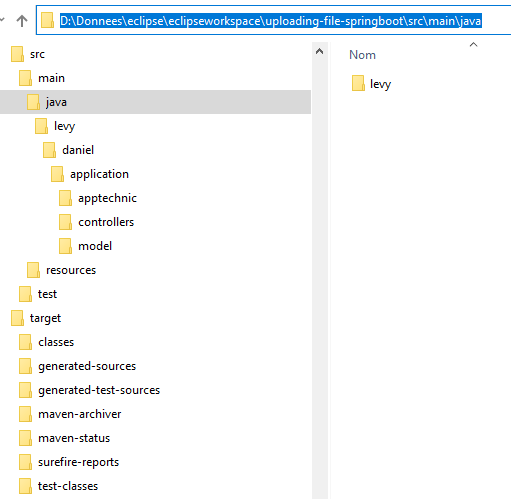
**Ce répertoire des classes compilées doit être intégré dans le classpath de la ligne de commande.**

### Répertoire des sources

Une différence essentielle avec la ligne de commandes du RUN est que le DEBUG doit **accéder aux sources des classes Java** pour s'y déplacer.

Il faut **rajouter le chemin de la racine des sources dans le classpath** :

D:\Donnees\eclipse\eclipseworkspace\uploading-file-springboot\src\main\java dans l'exemple



**Ce répertoire des sources doit être intégré dans le classpath de la ligne de commande.**

### Classpath intégrant toutes les bibliothèques utilisées (jar)

Un classpath se déclare dans une ligne de commande DEBUG Java avec -Djava.class.path= suivi de tous les chemins des classes compilées, des sources Java, et de toutes les bibliothèques (jar), le tout entre double-quotes ".

La ligne de commande devient donc à ce stade :

jre=%path\_jdk%\jre, -Dfile.encoding=UTF-8, -Djava.class.path="%PATH\_PROJET\_uploading-file-springboot%\target\classes; %PATH\_PROJET\_uploading-file-springboot%\src\main\java;*tous les chemins de tous les jar*"

On y trouve bien :

1. La JRE du JDK, l'encodage
2. l'ordre -Djava.class.path=,
3. Puis dans le classpath (entre double-quotes) : la référence aux classes compilées
4. La référence aux sources
5. et la référence à tous les jars.

* **Le classpath est mis entre double-quotes "** au cas où il y aurait des espaces dans le chemin des jar.
* **Les chemins des jar sont séparés par des point-virgule** ";".

On utilise le fichier dans Notepad "ligne\_commande\_debug.txt" pour récupérer les chemins des jars.

### Résumé de la syntaxe de la ligne de commandes du DEBUG

[JRE du JDK], -Djava.class.path="[chemin classes];[chemin sources];[tous chemins de tous jars]"

## Modifier la ligne de commande original d'ECLIPSE

### Retirer les ajouts d'ECLIPSE

Eclipse rajoute pour son débogage l'instruction :

-agentlib:jdwp=transport=dt\_socket,suspend=y,address=localhost:63044 …

Il faut donc retirer ces instructions dans l'éditeur RUN de Notepad++

Le début de la ligne de commande originale :

"C:\Program Files\Java\jdk1.8.0\_231\bin\javaw.exe" -agentlib:jdwp=transport=dt\_socket,suspend=y,address=localhost:50361 -javaagent:D:\Donnees\eclipse\eclipse\_2018\_12\_4\_10\eclipse\configuration\org.eclipse.osgi\405\0\.cp\lib\javaagent-shaded.jar -Dfile.encoding=UTF-8 -classpath

Devient donc :

"C:\Program Files\Java\jdk1.8.0\_231\bin\javaw.exe" -Dfile.encoding=UTF-8 -classpath

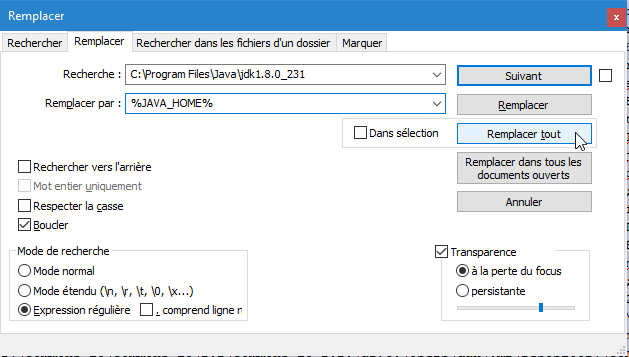
### Substituer les variables d'environnement (Local Paths) aux chemins en dur

**Objectif** : Cette étape optionnelle a l'intérêt de simplifier la ligne de commande résultante et d'améliorer la maintenabilité de la ligne de commande sur différentes machines. L'utilisation des variables d'environnement permet en effet de s'affranchir de la localisation réelle sur chaque machine du JDK, du dépôt MAVEN, …

Les variables d'environnement **s'utilisent toujours comme en MS-DOS entourées de %**.

On substitue donc aux chemins en dur leurs variables d'environnement dans l'éditeur DEBUG de Notepad ++.

#### Substituer %JAVA\_HOME% à C:\Program Files\Java\jdk1.8.0\_231



#### Substituer jre=%JAVA\_HOME%\jre à "%JAVA\_HOME%\bin\javaw.exe"

La ligne dans l'éditeur DEBUG de NotePad++ doit commencer par jre=%JAVA\_HOME%\jre, sans guillemets.

Bien mettre le séparateur virgule "," à la fin.

#### Rajouter une virgule après Dfile.encoding=UTF-8

jre=%path\_jdk%\jre, -Dfile.encoding=UTF-8,

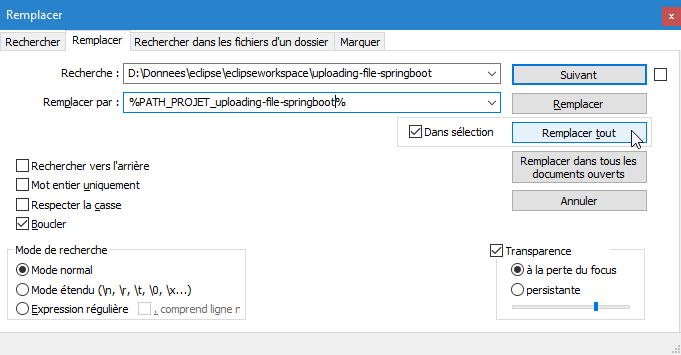
#### Substituer -Djava.class.path= à -classpath

jre=%JAVA\_HOME%\jre, -Dfile.encoding=UTF-8, -Djava.class.path=

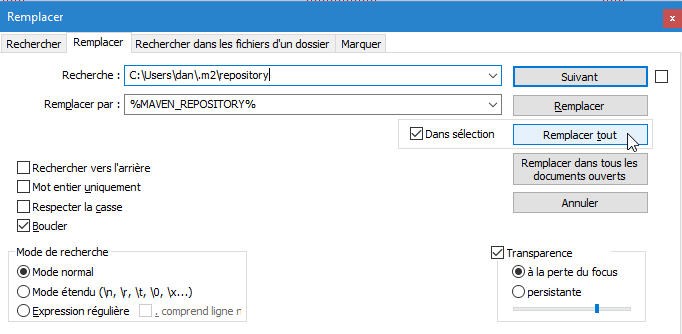
#### Mettre tous le classpath (jars) entre quotes

Les quotes " " permettent de s'affranchir des cas où il y a des espaces dans les chemins des jars.

#### Substituer %PATH\_PROJET\_uploading-file-springboot% à D:\Donnees\eclipse\eclipseworkspace\uploading-file-springboot



#### Substituer %MAVEN\_REPOSITORY% à C:\Users\dan\.m2\repository



#### RAJOUTER LE CHEMIN DES SOURCES %PATH\_PROJET\_uploading-file-springboot%\src\main\java dans le classpath

jre=%JAVA\_HOME%\jre, -Dfile.encoding=UTF-8, -Djava.class.path="%PATH\_PROJET\_uploading-file-springboot%\target\classes;%PATH\_PROJET\_uploading-file-springboot%\src\main\java;