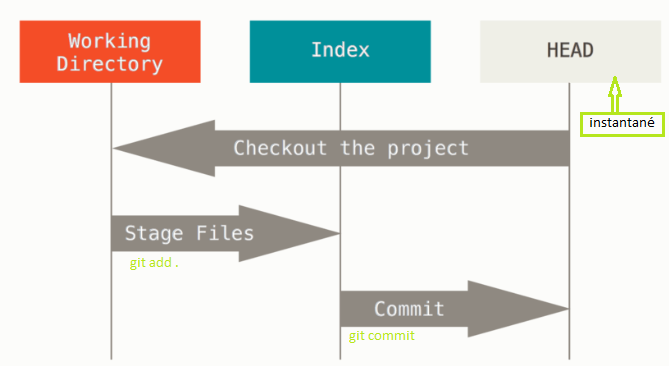
# Git : Gestionnaire de contenu

* Git contient trois instantanés :

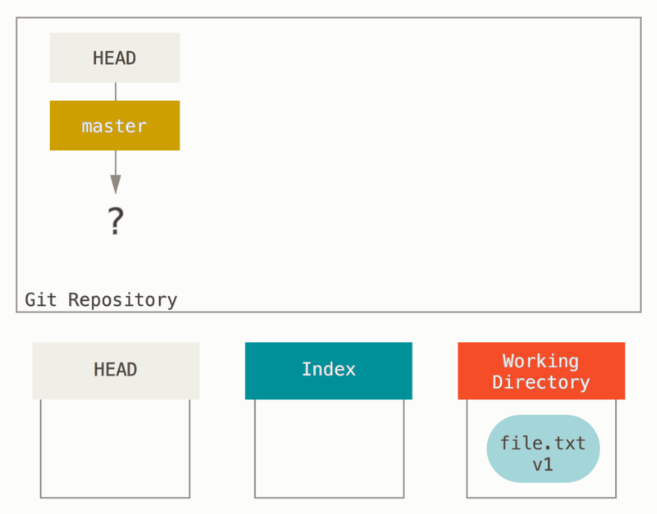
HEAD : est un pointeur sur la référence de la branche actuelle, qui est à son tour un pointeur sur le **dernier commit réalisé sur cette branche**. Le HEAD comme l’instantané de votre dernière validation (Dernier commit validé).

L’indexe : est votre **prochain commit proposé**. « zone de préparation » de Git du fait que c’est ce que Git examine lorsque vous lancez git commit.

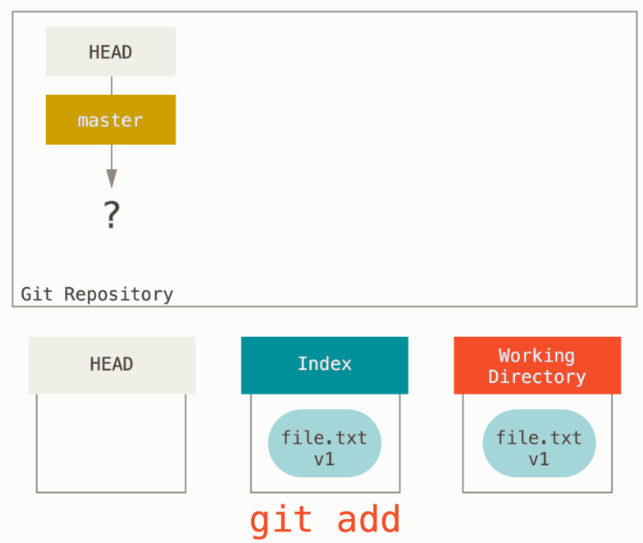
Le répertoire de travail : La copie du travail.



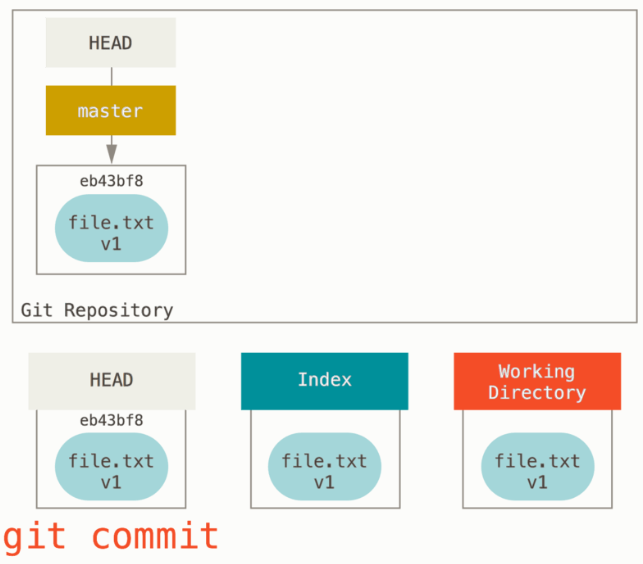
git init : Crée le dépôt Git avec une référence HEAD qui pointe sur la branche à naitre :



git add . : prend le contenu du répertoire de travail et le copie dans l’index.

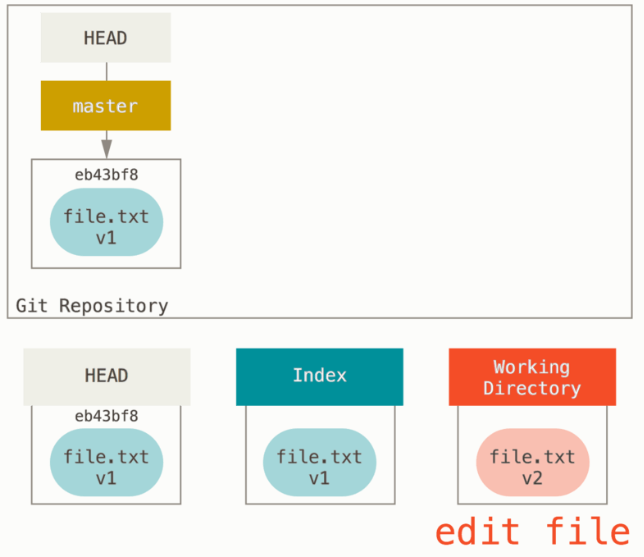


git commit : prend le contenu de l’index et le sauve comme un instantané permanent (le sauve dans le HEAD), crée un objet commit qui pointe sur cet instantané (Pointeur de HEAD sur la dernière validation) et met à jour master pour pointer sur ce commit.

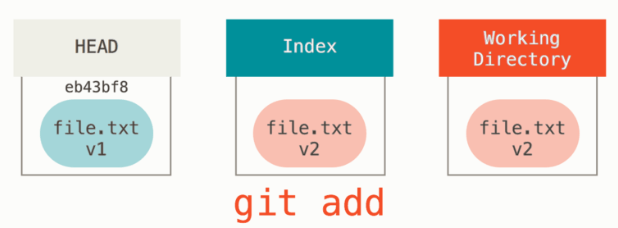


git status : vérifie les modifications entre les trois arborescences.

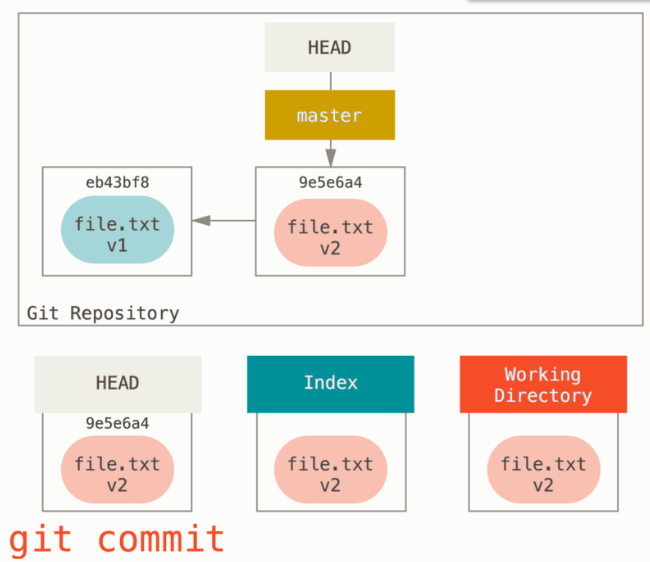
Ont faisant une modification sur le fichier v1 sur notre répertoire de travail (appelant cette version de fichier v2).



Si nous lançon git status => nous verrons le fichier en rouge comme « Modifications qui ne seront pas validées » car cette entrée est différente entre l’index et le répertoire de travail. Ensuite, nous lançons git add dessus pour le monter dans notre index.



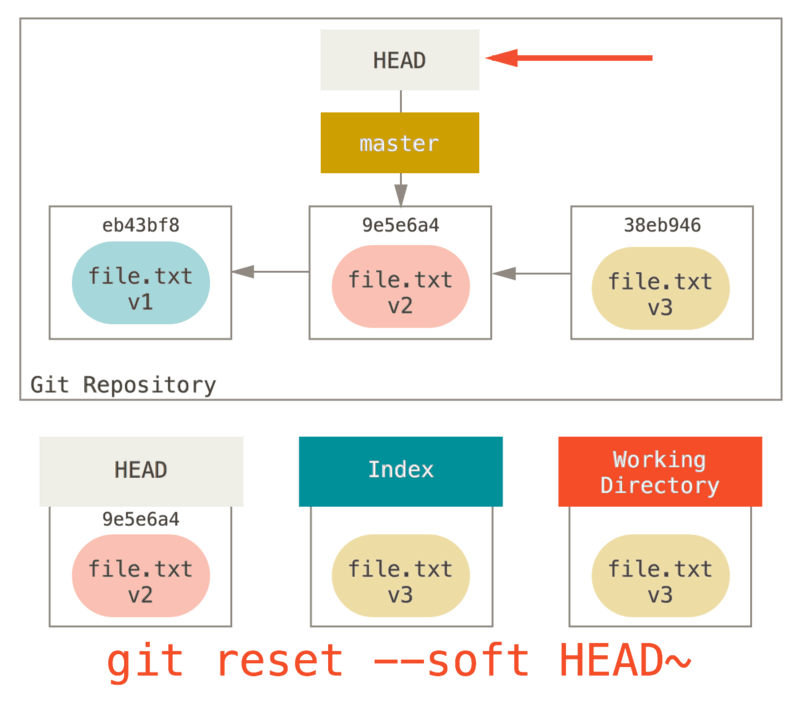
À ce point, si nous lançons git status, nous verrons le fichier en vert sous « Modifications qui seront validées » parce que l’index et HEAD diffèrent, c’est-à-dire que notre prochain **commit** proposé est différent de notre dernier **commit**. Finalement, nous lançons git commit pour finaliser la validation.



Maintenant, git status n’indique plus rien, car les trois arborescences sont à nouveau identiques

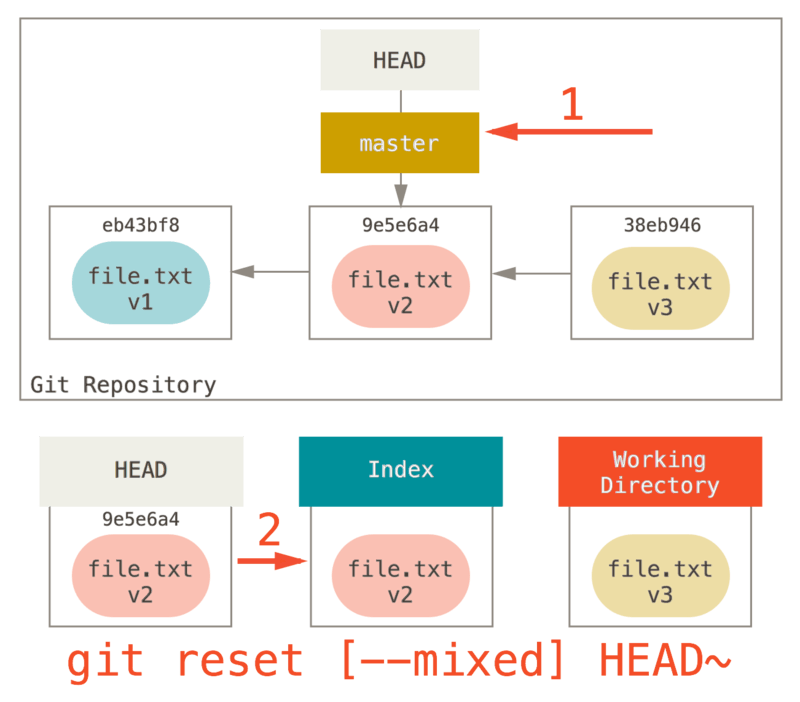
* Différence entre checkout et reset :
* Reset (Déplace la branche que HEAD pointe) :

git reset --soft HEAD~ : On fait un git reset sur HEAD, ainsi, la branche sera replacé où elle était (c’est à dire un pas en arrière) sans changer ni l’index ni la copie de travail.



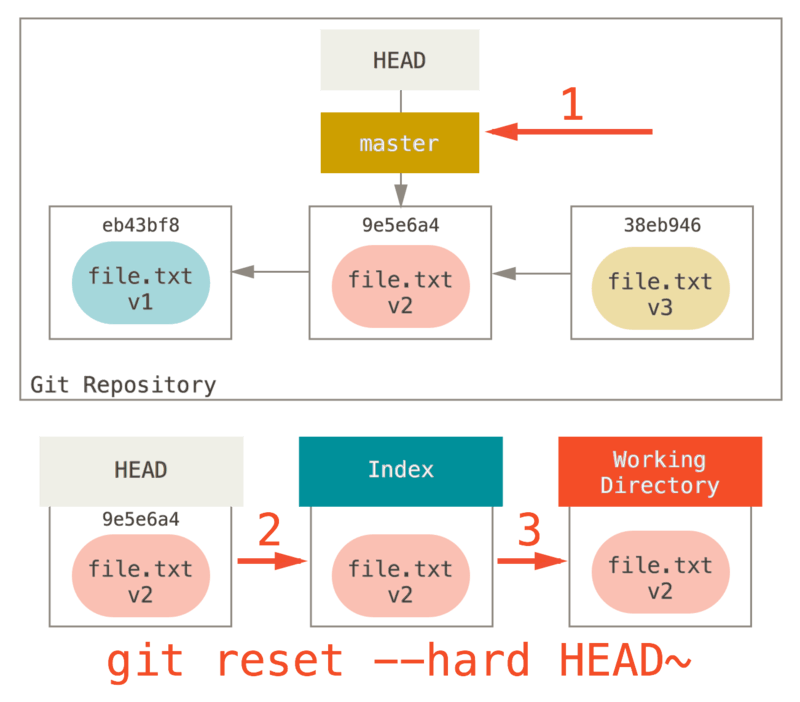
Si vous lancez git status, vous verrez en vert la différence entre l’index et le nouveau HEAD.

git reset --mixed HEAD~ : Mise à jours de l’index avec le contenu de l’instantané pointé par HEAD.



Vous êtes revenu à l’état précédant vos commandes git add et git commit

git reset --hard HEAD~ :



Récapitulatif :

La commande reset remplace ces trois arbres dans un ordre spécifique, s’arrêtant lorsque vous lui indiquez :

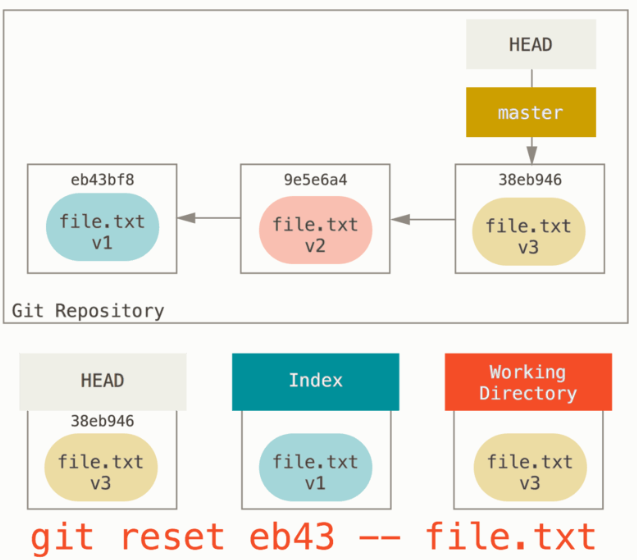
1. Déplace la branche pointée par HEAD **(s’arrête ici si --soft)**
2. Fait ressembler l’index à HEAD **(s’arrête ici si --mixed)**
3. Fait ressembler le répertoire de travail à l’index (si --hard).

git reset file.txt : c’est un raccourcis de git reset --mixed HEAD file.txt qui va

1. déplacer la branche pointée par HEAD **(sauté)**
2. faire ressembler l’index à HEAD **(s’arrête ici)** pour le fichier concerné

Donc, en substance, il ne fait que copier file.txt de HEAD vers index. Cette commande va désindexer le fichier.txt

Git reset sha file.txt : va faire la même chose, copier file.txt dans le commit du hash dans l’index.



* Checkout

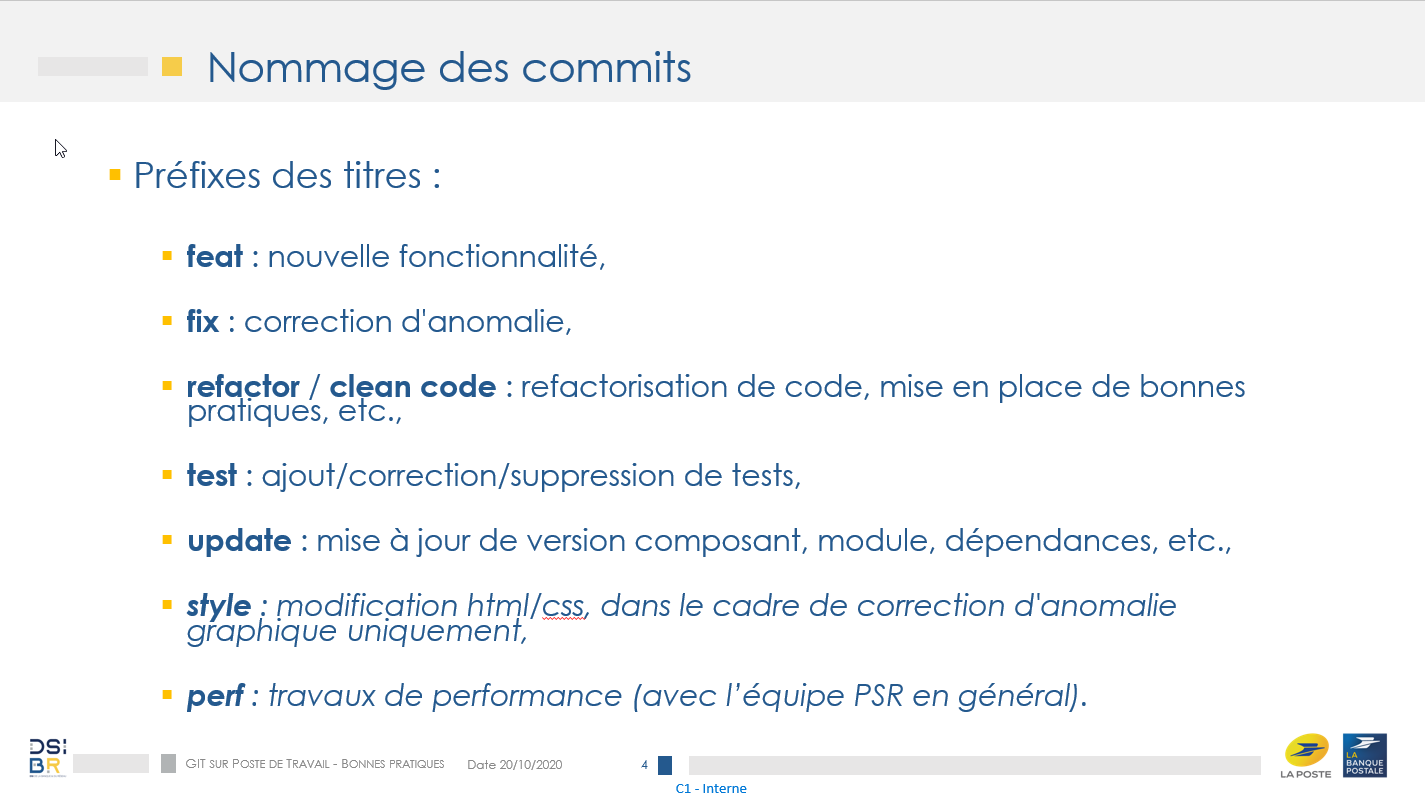
git checkout [branche] : similaire à git reset --hard [branch] il met à jours les trois arborescences avec deux différences majours.

Checkout preserve le répertoire de travail et ne casse rien. Il essaie de faire une fusion simple dans le répertoire.

Les commandes Git :

|  |  |
| --- | --- |
| git status |  |
| git pull |  |
| git push |  |
| git add . |  |
| git commit –m ’’….’’ |  |
| git rebase –i hashCommit |  |
| git config --list |  |
| git branche nomBranch |  |
| git checkout –b nomBranch |  |
| git reset HEAD remoteBranch |  |
|  |  |

Naming des commits :



* Pousser les évolutions de code de la branche locale au remote :

## Pousser les modifs dans ta branche local :

Git add .

Git commit –m ’’….’’

Git push

## Faire un checkout sur la branche remote:

Git checkout remote\_branche

## Pousser les modifications de la branche local vers la branche remote

Git pull

Git pull origin branche\_locale

Git push

* Récupérer le code de la branche remote à la branche local (merge/rebase)
* merge

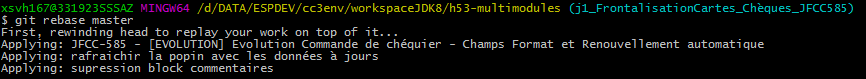
Git checkout branche\_local

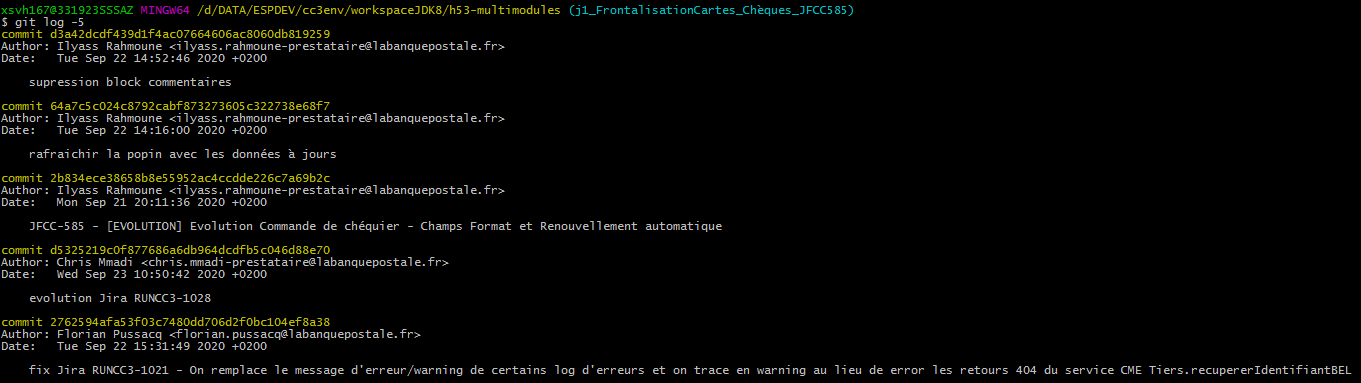
Git merge branche\_remote

* rebase

1. Depuis ta branche local, pense à faire git rebase par rapport à la branche master (Remote) :

$git rebase master -> git va mettre les commits de ma branche à part, il va récupérer les commits du master, les injecter à partir l’ancien commit en commun entre ma branche locale et la branche remote, et rajouter mes nouveaux commits en dessus de tous les commits.





1. Fusionner les commits en un seul commit :

Rebase sur un ensemble de commit et les fusionner en un seul commit avec renommage :

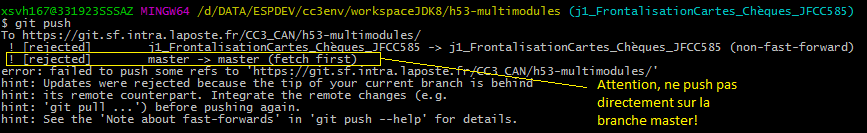
Git rebase -i d5325219c0f877686a6db964dcdfb5c046d88e70

Modifier « pick » par “r” ou “s” ou …

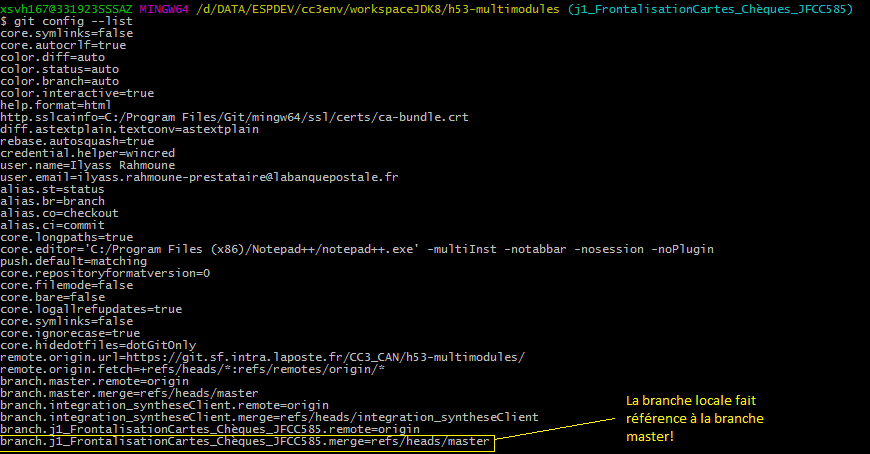
Modifier le rire du message en cas de « r »

Les commits préfixé par « squash » seront fondu dans le commit précédents

1. Attention, ne push pas depuis la branche locale à la branche master :



Ce problème vient du fait que la branche locale pointe sur la branche master en remote :$git config --list :

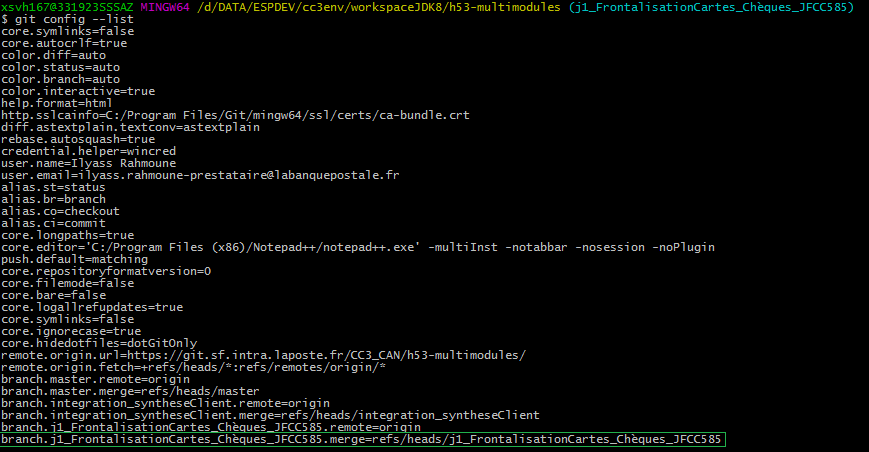


Pour résoudre ce problème, exécute la commande git :

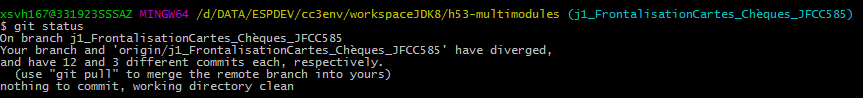
$ git branch -u origin/j1\_FrontalisationCartes\_Chèques\_JFCC585

(Branch j1\_FrontalisationCartes\_Chèques\_JFCC585 set up to track remote branch j1\_FrontalisationCartes\_Chèques\_JFCC585 from origin.)

Ainsi :



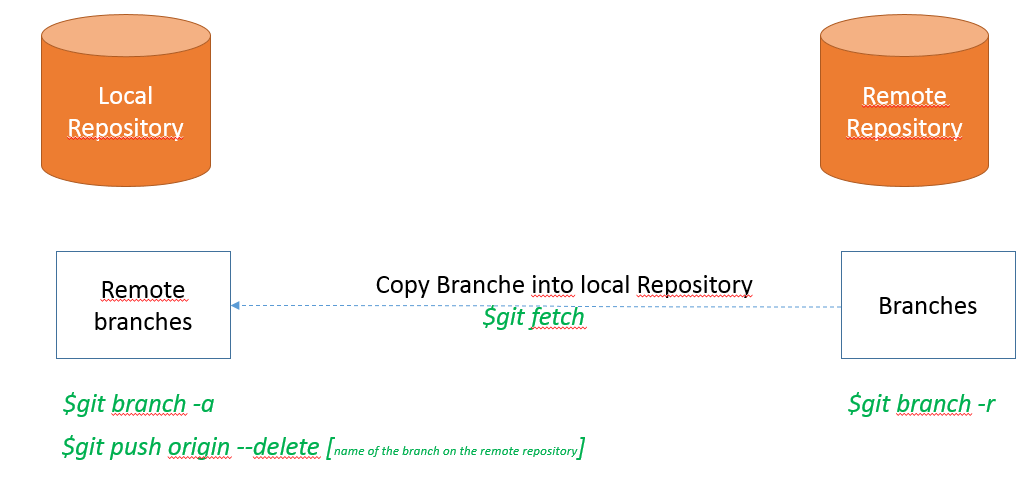
Et on peut faire le push correctement sans erreurs :



* Lister les branches locales et remotes :

*$git branch –a* : liste les branches remotes qui existe dans le repo local

*$git branch –r* : liste les branches du remote repository.



* Dupliquer une branche remote de Git :

$git push [REMOTE NAME] [WHERE SHOULD THE REMOTE BRANCH POINT TO]:[REMOTE BRANCH NAME]

Exemple:

*$git push origin 125afe4:master-copy*

|  |
| --- |
| $git commit –am ‘’message commit’’ ⬄ $git add . | $git commit –m ‘’message commit’’ |

|  |  |  |
| --- | --- | --- |
| |  | | --- | | *$git checkout –b nom\_branche* |  |  | | --- | | *$git branch nom\_branche*  *$git checkout nom\_branche* |   ⬄ |

Pour récupérer les commits de la branche master

* Différence entre master, origin et origin/master :

master : local branch

origin/master : remote branch, the local copy of the master branch in the remote.

origin: it means remote.

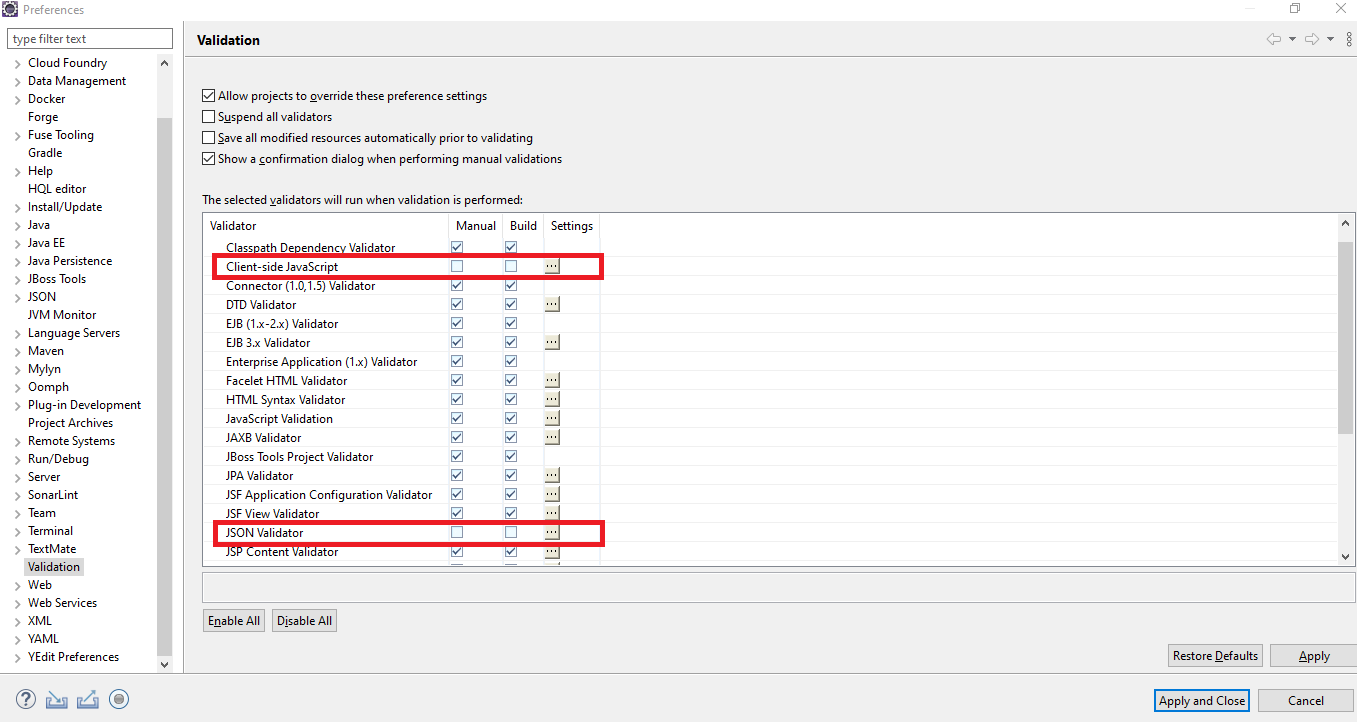
# Eclipse Netwok connection : Pondor Password

UserName/Password : xsvh167 / Labanquepostale\_10

2.1 Stabiliser ton environnement Eclipse:

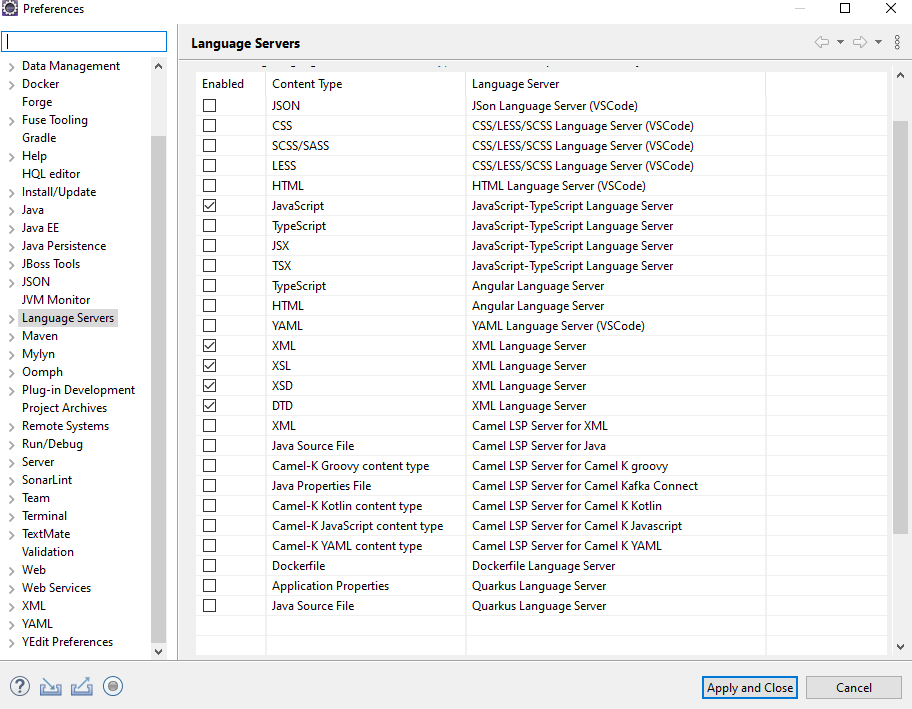
2.1.1 Les types validations Eclipse:

Dans l’onglet problems d’éclipse, les problèmes sont triés par type d’erreur. Dans la colonne type, c’est le type de validation qui pose problème (Exemple: JSON Problem). Ainsi, il faut désactiver de type validation dans Window -> Preference -> Validation:



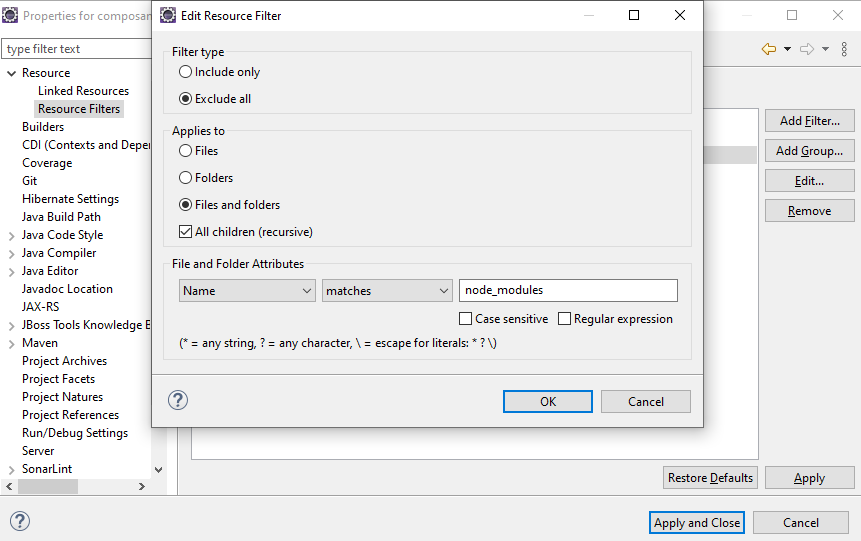
2.1.2 Editeur de Code : Window -> Preference -> Language Servers : Customize Edit code in Eclipse

Décoche les éditeurs de code comme JSON/HTML/CSS… comme ci-dessous:



2.1.3 Ajouter les ressources filter: Il faut exclure les node\_modules de dans tous les projets:

Clique droit sur un projet -> properties -> Resource -> Resource Filter -> Add Filter :



2.2. Maven Project Build Lifecycle Mapping problem:

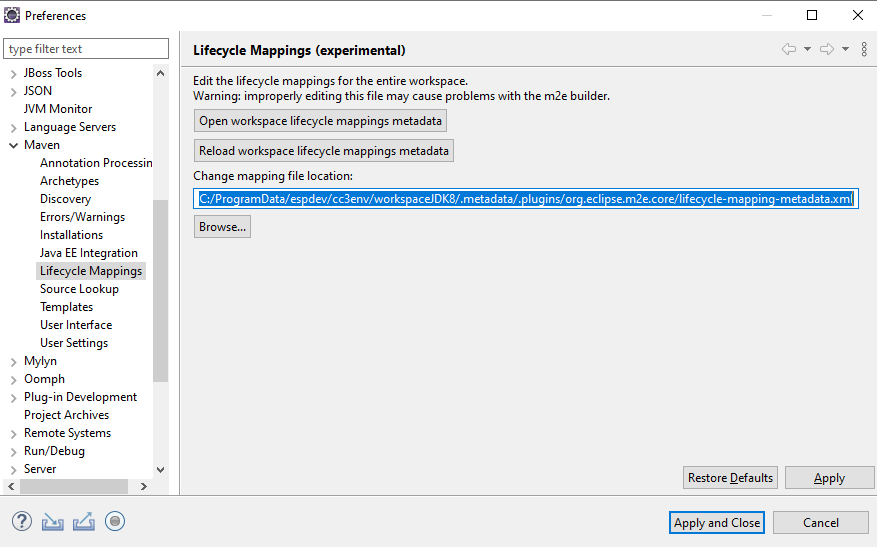
Ajouter le fichier C:/ProgramData/espdev/cc3env/workspaceJDK8/.metadata/.plugins/org.eclipse.m2e.core/lifecycle-mapping-metadata.xml:

Et cliquer sur le bouton reload workspace lifecycle mappings metadata:



Contenu du fichier:

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <lifecycleMappingMetadata>  <pluginExecutions>  <pluginExecution>  <pluginExecutionFilter>  <groupId>org.codehaus.mojo</groupId>  <artifactId>exec-maven-plugin</artifactId>  <versionRange>1.6.0</versionRange>  <goals>  <goal>exec</goal>  </goals>  </pluginExecutionFilter>  <action>  <ignore />  </action>  </pluginExecution>  </pluginExecutions>  </lifecycleMappingMetadata> |



# LBP : Outil pour appliquer les commandes git sur tous les projets en une ligne de commande.

## Vérifier l’installation : lbp –help

## Mise à jours des pom.xml des projets locaux :

lbp po –s

## Pull tous les projets en une ligne de commande :

lbp cmd git pull

## Reset Hard de tous les projets p/p à une branche :

lbp cmd reset –hard origin/integration\_0\_x

## Modification de l’utilisateur :

lbp uid ‘TEER025’

lbp urls

lbp uid

NB: « lbp command/cmd » permet d’éxecuter une commande dans tous les projets.

# NPM: gestionnaire des dépendances dans node.

## Installer les dépendances d’un projet :

npm i

## Démarrer un projet node :

npm start

NB : pour démarrer tous les projets Node, en utilise ConEmu. (ConEmu génère les fichiers \*bundle.js).

Copie node\_module d’un projet vers un autre (dans le cas on n’arrive pas à installer node\_module via npm à cause d’un problème de proxy) :

$ cp -r ../composants-applicatifs-DossierClientNumerique/node\_modules/ .

# Démarrage du projet en local :

## Démarrer les deux modules en mode débug : H54 et H55.

## Lancer ConEmu pour générer les bundles de tous les composants SPA.

## Lancer l’application en local :

* Dans le chemin « D:\DATA\ESPDEV\CC3Links-master » . Lancer index.htm
* Taper idClient et kle Bureau rdv et cliquer sur « Go ! » ☺

# SecurityBouchonConfig.xml :

Rooting Roles : droit d’accès à une application (Exemple : Droit d’accès à l’application vente au moment de débranchement).

Security Roles : droit utilisateur, qu’est-ce qu’il a comme droit à faire.

Analyse des logs : Splunk

# Requête splunk :

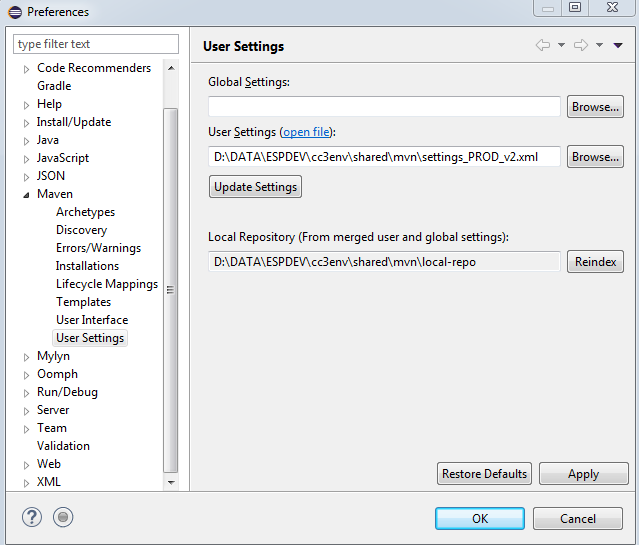
Copier la requête dans le fichier ReadMe dans C:\Users\xsvh167\Desktop\Splunk Script.

# Repository Nexus : User Setting

Purge dependecy local reposirtory command for maven :

Right click on the project -> Run as -> Maven build… -> Goals : and right the command ***dependency :purge-local-repository***

Téléchargement des dépendances par Maven (Autorisation de l’accès au repo Nexus pour télécharger les jars. Exemple : Jar Toolbox 3.0)



Dans window->preferences->Maven->User Settings

Clique sur Open files.

Le paramétrage du user pour accéder au repos Nexus est fait dans la balise:

<servers>

<server>

<id>DISFE\_CC3\_Central</id>

<username>xipx968</username>

<password>@Zerty10</password>

</server>

</servers>

Au début de chaque nouveau mois, on change le password avec suffixe le mois en cours, exemple pour le mois d’octobre, le password est @Azerty10.

# Difference entre requête http et requête HTML

http://localhost:7005/personnalisationformulesdecompte/templates/personnalisationformuledecompte.compiled.html?cf4dc294afb7b26193e1532f2a

La requête ci-dessus est une requête qui permet de charger le code HTML (.comple.html). ci-dessous une vrai requête http : http://localhost:7005/api/clients/021024590/offres/R2-V423-OFR0887/?isMajeur=true&isResidentFrancais=true&situationMetier=VENTE&bureauRdvs

API Map {url, key} : Définit le contexte de routage en mode bouchonné.

ApiMapBouchonConfig.xml : Document qui définit les bouchons (profils utilisateurs) remplaçant le système d’assemblage.

gcp.propoerties : fichier properties ou le mode bouchonné est paramétré (ASSEMBLAGE\_API\_MAP\_BOUCHON=true pour activer les bouchons => développement sans besoin du système d’assemblage).

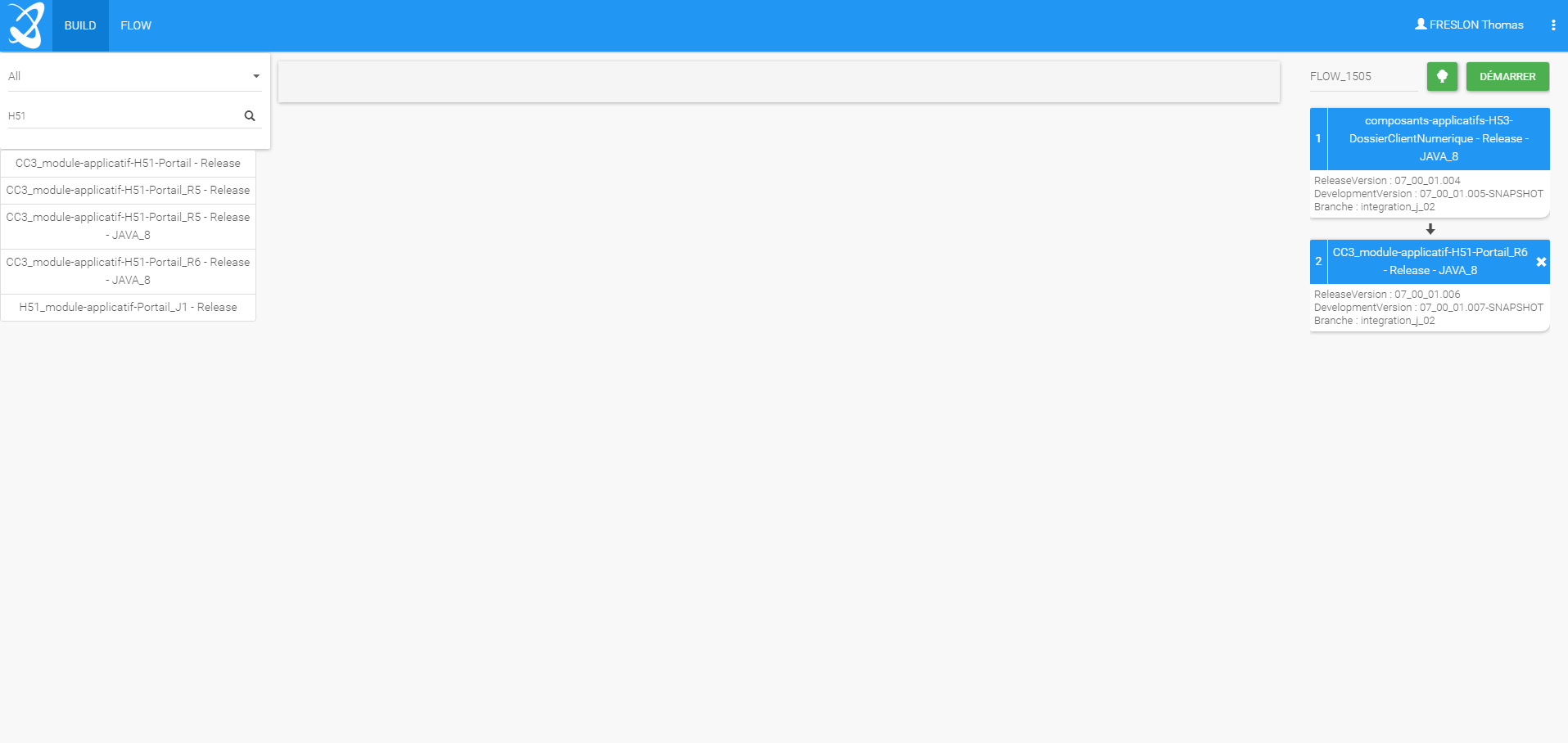
# Build et Deploiement

Dans l’ADB, choisir ton composant à builder (Dossier Client Nulmérique) ; Ajoute le au build

Dans l’ADB, choisir le module qui contient ce composant (H51) ; Ajoute le au build (Integration\_j\_02).

Dans la vue de build de H51, choisir la dernière version de DCNn qui vienne d’être générer pour builder dans l’étape précédente.

Démarrer le build

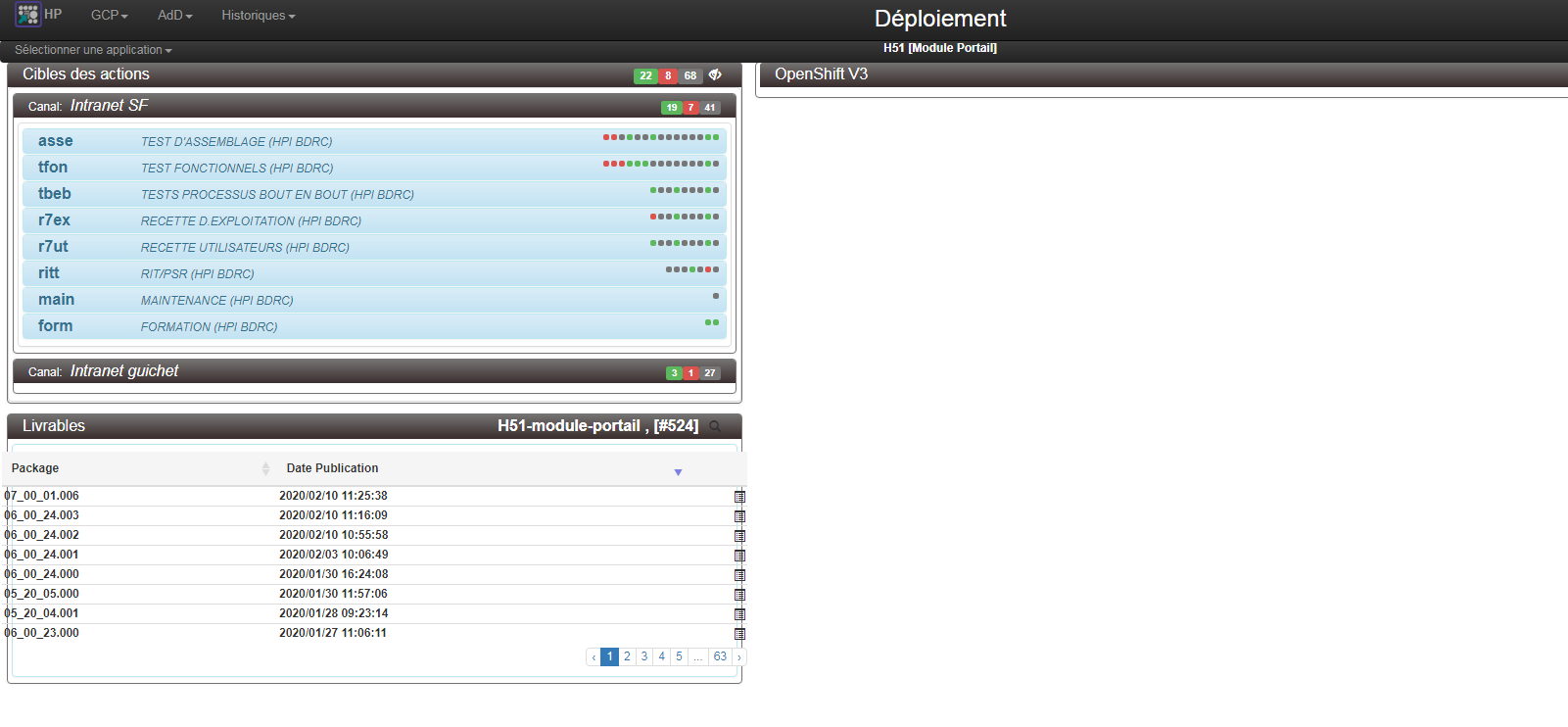


Déploiement :

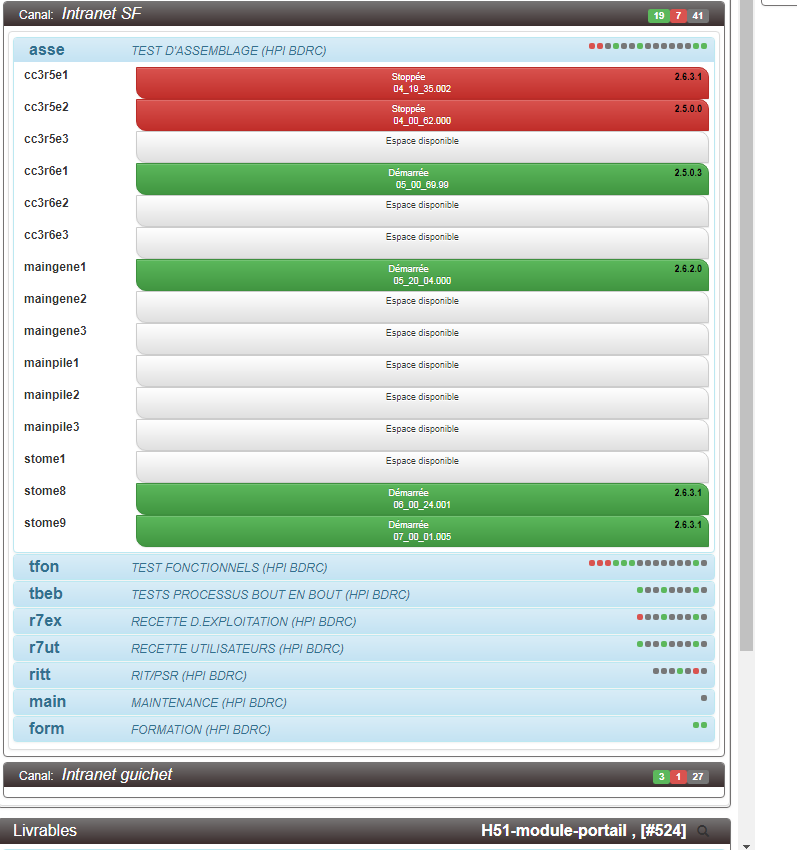
Dans la plateform ADD (<https://add-hp.stomv2.admin.hpmetier.sf.intra.laposte.fr/ad_/broker>)

Dans l’onglet AdD, choisir déploiement dans l’onglet, en suite, séléctionner H51 dans la liste des applications affichées.

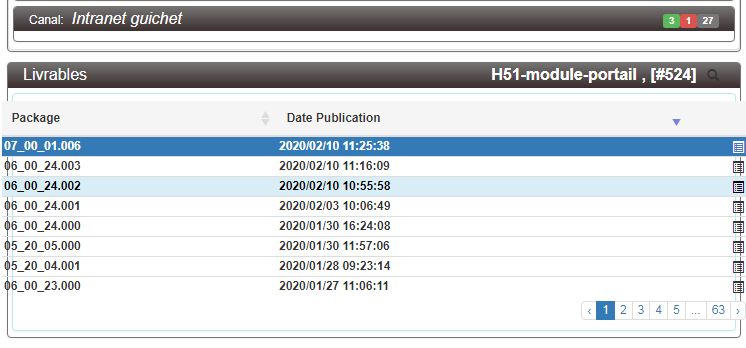
Dans la fenêtre < Canal : Intranet SF > , choisir l’un des environnement de déploiement(ex : asse pour l’environnement TAS (Test d’ASsemblage)).



Par la suite, choisir le <stome9> pour en deployer.



Descendre vers l’onglet ‘Livrables’ le bas pour choisir la dernière version du module H51 builder :



Ensuite clique sur Déploiement dans l’écran <Actions sur les Application> :



MUI : Mise en production exceptionnelle.

Tools / Stack techniques

|  |  |
| --- | --- |
|  | Outils/Framework de dévloppement |
| ToolBox | Fichiers script JS + fichiers de style CSS importer dans le header d’une template pour son style et les animations. |
| Grunt | Outil de création des tâches automatisées à bas du langage JavaScript. Il compile tous les fichiers JS et genère un seul fichier bundle.js qui fusionne tous les fichiers JS. |
| Lombok | Framework qui génère les constructeurs, getters/setters et toString(). |
| ConEmu | Crée les fichiers \*bundles.js du SPA.  Exemple de commandes de runGrunt:   |  | | --- | | > -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-VentePanier -cur\_console:t:ventePanier\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd"  -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-DossierClientNumerique -cur\_console:t:DCN\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd"  -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-PersonnalisationCarte -cur\_console:t:persoCarte\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd"  -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-PersonnalisationCompte -cur\_console:t:persoCompte\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd"  -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-InterrogationsRisque -cur\_console:t:interroRisque\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd"  -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-SouscriptionLDD -cur\_console:t:persoLDD\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd"  -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-PropositionContrat -cur\_console:t:propositionContrat\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd"  -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-PersonnalisationCommun -cur\_console:t:persoCommun\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd"  -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-Portail -cur\_console:t:portail\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd"  -cur\_console:d:D:\DATA\ESPDEV\cc3env\workspaceJDK8\composants-applicatifs-Parapheur -cur\_console:t:parapheur\_JDK8 D:\DATA\ESPDEV\cc3env\scripts\runGrunt.bat /k "%ConEmuBaseDir%\CmdInit.cmd" | |
| Splunk | Outil d’analyse des logs. |
|  |  |
|  | Outils de gestion de projet |
| RTC | Outil de remontée d’anomalies détectées. |
| Post It | Création des tickets pour les demandes d’accès aux outils (JIRA/Git …) |

FwAD

+ Le controlleur permet l’injection des dépendances.

+ Dans le module, on déclare les constantes, directives, et services … à injecter dans le controleur.

+ Chaque nouveau module doit être déclaré dans le module père main.module.

Les versions dans les POMs :

* Vente Panier

<description>Composant Applicatif VentePanier</description>

<properties>

<composant-applicatif-rechercheClient.adb.version>05\_00\_54.000</composant-applicatif-rechercheClient.adb.version>

<composant-applicatif-commun.adb.version>05\_00\_62.000</composant-applicatif-commun.adb.version>

<composant-applicatif-ecransgeneriques.adb.version>05\_00\_54.000</composant-applicatif-ecransgeneriques.adb.version>

<composant-applicatif-entreeenrelation.adb.version>05\_00\_54.000</composant-applicatif-entreeenrelation.adb.version>

<fwmc.adb.version>4.4.31</fwmc.adb.version>

<fwad.adb.version>3.8.10</fwad.adb.version>

<composant-applicatif-finalisationEntretien.adb.version>05\_00\_62.000</composant-applicatif-finalisationEntretien.adb.version>

<toolbox.adb.version>3.0.33</toolbox.adb.version>

</properties>

* Module Personnalisation

<name>CC3\_ModuleApplicatif\_H55-SouscriptionLDD</name>

<properties>

<composant-applicatif-personnalisationCompte.adb.version>05\_00\_62.001-SNAPSHOT</composant-applicatif-personnalisationCompte.adb.version>

<toolbox.adb.version>3.0.33</toolbox.adb.version>

<composant-applicatif-personnalisationformuledecompte.adb.version>05\_00\_62.001-SNAPSHOT</composant-applicatif-personnalisationformuledecompte.adb.version>

<composant-applicatif-personnalisationCommun.adb.version>05\_00\_63.001-SNAPSHOT</composant-applicatif-personnalisationCommun.adb.version>

<composant-applicatif-propositionContrat.adb.version>05\_00\_62.001-SNAPSHOT</composant-applicatif-propositionContrat.adb.version>

<fwad.adb.version>3.8.10</fwad.adb.version>

<composant-applicatif-personnalisationautreproduitAQ.adb.version>05\_00\_62.001-SNAPSHOT</composant-applicatif-personnalisationautreproduitAQ.adb.version>

<composant-applicatif-souscriptionLDD.adb.version>05\_00\_62.001-SNAPSHOT</composant-applicatif-souscriptionLDD.adb.version>

<fwmc.adb.version>4.4.42</fwmc.adb.version>

<composant-applicatif-personnalisationCarte.adb.version>05\_00\_62.001-SNAPSHOT</composant-applicatif-personnalisationCarte.adb.version>

</properties>

* Module MultiVente

<name>CC3\_ModuleApplicatif\_H54-EERMultivente</name>

<properties>

<composant-applicatif-rechercheClient.adb.version>05\_00\_54.000</composant-applicatif-rechercheClient.adb.version>

<composant-applicatif-commun.adb.version>05\_00\_54.000</composant-applicatif-commun.adb.version>

<composant-applicatif-entreeenrelation.adb.version>05\_00\_54.000</composant-applicatif-entreeenrelation.adb.version>

<fwmc.adb.version>4.4.31</fwmc.adb.version>

<fwad.adb.version>3.8.10</fwad.adb.version>

<composant-applicatif-ventePanier.adb.version>05\_00\_63.006-SNAPSHOT</composant-applicatif-ventePanier.adb.version>

<composant-applicatif-finalisationEntretien.adb.version>05\_00\_62.000</composant-applicatif-finalisationEntretien.adb.version>

<toolbox.adb.version>3.0.33</toolbox.adb.version>

</properties>

Jalon 2 (J2) : VAD (Vente à Distance)

Les spécifications fonctionnelles sont faites par le MOA.

Parapher : Composant pour la signature électronique.

DCN(Dossier Client Numérique) : Composant qui gère les documents dématérialisés du client.

JAVA 8 :

The proposed answers are great. Just would like to suggest an improvement to handle the case of null list using Optional.ofNullable, [new feature in Java 8](https://docs.oracle.com/javase/8/docs/api/java/util/Optional.html):

List<String> carsFiltered = Optional.ofNullable(cars)

.orElseGet(Collections::emptyList)

.stream()

.filter(Objects::nonNull)

.collect(Collectors.toList());

So, the full answer will be:

List<String> carsFiltered = Optional.ofNullable(cars)

.orElseGet(Collections::emptyList)

.stream()

.filter(Objects::nonNull) //filtering car object that are null

.map(Car::getName) //now it's a stream of Strings

.filter(Objects::nonNull) //filtering null in Strings

.filter(name -> name.startsWith("M"))

.collect(Collectors.toList()); //back to List of Strings

Portail:

1. Gcp.properties:

ASSEMBLAGE\_HTTP\_URL\_BASE=http://h50-intranetsf-slot1-asse-stome9-unsecured.service-dev.stomv2.hpmetier.sf.intra.laposte.fr

1. SecurityBouchonConfig.xml

<bean id=*"userDetailsServiceBouchon"* class=*"fr.laposte.disf.fwmc.rest.security.bouchon.CustomUserDetailsServiceBouchon"*

init-method=*"initFromXml"*>

<property name=*"userId"* value=*"TCCA011"* />

<property name=*"userDetails"*>

<ref bean=*"userProfileBouchonTEER001"* />

</property>

</bean>

1. COFI\_TEER001.xml

<!-- Utilisateur -->

<bean id=*"userProfileBouchonTEER001"* class=*"fr.laposte.disf.fwmc.rest.security.bouchon.ConfigurableUserProfileBouchon"*>

<property name=*"userDN"* value=*"uid=TCCA011,ou=utilisateurs,o=SF,o=la poste,c=fr"*/>

<property name=*"name"* value=*"COFI"*/>

<property name=*"firstName"* value=*"TCCA011"*/>

<property name=*"businessProfiles"*>

<list>

<bean class=*"fr.laposte.disf.fwmc.arch.context.reference.impl.ConfigurableBusinessProfile"*>

<property name=*"code"* value=*"COFI"*/>

</bean>

</list>

</property>

<property name=*"workstation"*>

<bean class=*"fr.laposte.disf.fwmc.arch.context.reference.impl.ConfigurableWorkstation"*>

<property name=*"code"* value=*""*/>

</bean>

</property>

<property name=*"authentifiedWorkstation"* value=*"false"*/>

<property name=*"executiveEntity"*>

<bean class=*"fr.laposte.disf.fwmc.arch.context.reference.impl.ConfigurableEntity"*>

<property name=*"code"* value=*"930530"*/>

<property name=*"type"* value=*"Bureau"*/>

</bean>

</property>

<property name=*"changePwdInterval"* value=*"-570"*/>

<property name=*"authentificationLevel"* value=*"1"*/>

<property name=*"userAgent"* value=*"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36"*/>

<property name=*"username"* value=*"TCCA011"*/>

<property name=*"accountNonExpired"* value=*"true"*/>

1. Pom.xml

<composant-applicatif-dossierclientnumerique.adb.version>07\_00\_01.009-SNAPSHOT</composant-applicatif-dossierclientnumerique.adb.version>

Test RMOA :

|  |  |
| --- | --- |
| Environnement | User |
| R6 | TEER025  TTPB105 |
| J2 | TCRI001 |
| J1 | TCCM100 (client 118443445) |

Test TASS :

|  |  |
| --- | --- |
| Environnement | User |
| J2 | TCCA105  TCCA011 |
|  |  |

Les logs :

-Default performance : Logger du framework qui trace par défaut les paramètres d’appel d’une requête sauf son Body.

-Une template est mise en place qu'ils mettent à jours suite aux nouvelles préconisations :

* Les logs framework ont le niveau INFO
* le niveau des log pour les applis reste en Debug

Test unitaire :

@InjetcMocks : Seulement l’attribut de la classe sous test qui doit avoir l’annotation @injectMocks. Il fonctionne comme une sorte d’injection de dépendances pour le système testé.

@Mock : Toutes les autres dépendances doivent être annotés par @Mock.

@Spy : on l’utilise pour stuber les méthodes.

Il faut jamais combiner @InjectMocks et @Spy.

Classe de Test DossierCollecteService :

|  |
| --- |
| @RunWith(SpringJUnit4ClassRunner.**class**)  @ContextConfiguration(classes = {UtilConfig.**class**, KeyGeneratorConfig.**class**, NoSQLClusterConfig.**class**})  @WebAppConfiguration  **public** **class** DossierCollecteServiceTest {    @InjectMocks  DossierCollecteService dossierCollecteService;    /\*\* rolesService \*/  @Mock  IRolesSevice mockRolesService;    @Mock  ReferentielService mockReferentielService;    @SuppressWarnings("rawtypes")  @Mock  CMEService mockAppelsMetier;    @Mock  **private** SyntheseDossierCollecteMapping syntheseDossierCollecteMapping;    // Avec la nouvelle concetipon en bas, ce mock peut être viré, car on appel plus le mapper depuis la méthode tésté ici.  @Mock  DossierCollectePiecesJustificativesMapping dossierCollectePiecesJustificativesMapping;    MockMvc mockMvc;      @Autowired  **private** WebApplicationContext ctx;    @BeforeClass  **public** **static** **void** setUpFwmc() {  Framework.*start*();  }    @Before  **public** **void** setUp() {  **this**.mockMvc = MockMvcBuilders.*webAppContextSetup*(ctx).build();  MockitoAnnotations.*initMocks*(**this**);  } |

Service qui devrait être appelé par le contrôleur :

|  |
| --- |
| **public** List<DossierCollectePiecesJustificativesRessource> recupererListDossierCollecte(**final** String idClient) **throws** MetierException, IOException, ParseException {  List<DossierCollectePiecesJustificatives> dossierCollectePiecesJustificativeList = recupererListeDossiersCollectesParClientID(idClient);    // On appel le mapping par la suite  **Return** dossierCollectePiecesJustificativesMapping.mapFromList(dossierCollectePiecesJustificativeList);  } |

|  |
| --- |
| @SuppressWarnings("unchecked")  @Override  **public** List<DossierCollectePiecesJustificatives> recupererListeDossiersCollectesParClientID(**final** String idClient) **throws** IOException, MetierException, ParseException {    /\* List des dossiers de collectes finale à retourner à SPA \*/  List<DossierCollectePiecesJustificatives> dossierCollectePiecesJustificativeList = **new** ArrayList<>();    **try** {  VuesDossiersCollectesPiecesJustificatives vuesDossiersCollectesPiecesJustificatives = (VuesDossiersCollectesPiecesJustificatives) appelMetierService.recupererListDossiersCollectesPiecesJustificative(  VuesDossiersCollectesPiecesJustificatives.**class**, idClient);    /\* Filtrer les dossiers dont le statut est EnCours\_LBP \*/  List<DossierCollectePiecesJustificatives> dossiersCollectesEnCoursLBP = DossierCollectePredicates.*filtrerDossiersCollectes*(  vuesDossiersCollectesPiecesJustificatives.getVuesDossiersCollectesPiecesJustificatives(),  DossierCollectePredicates.*estEnCoursLBP*());    /\* Filtrer les dossiers dont le statut est EnCours\_LBP moins de soixante jours \*/  List<DossierCollectePiecesJustificatives> dossiersCollectesEnCoursLBPMoinsDeSoixanteJours = DossierCollectePredicates.*filtrerDossiersCollectes*(  dossiersCollectesEnCoursLBP,  DossierCollectePredicates.*estMoinsDeSoixanteJoursCalendaires*());    /\* Filtrer les dossiers dont le statut est Finalise ou Finalise\_LBP \*/  List<DossierCollectePiecesJustificatives> dossiersCollectesFinalisesOuFinalisesLBP = DossierCollectePredicates.*filtrerDossiersCollectes*(  vuesDossiersCollectesPiecesJustificatives.getVuesDossiersCollectesPiecesJustificatives(),  DossierCollectePredicates.*estFinaliseLBPOuFinalise*());    ArrayList<DossierCollectePiecesJustificatives> dossiersCollectesFusionCollection = **new** ArrayList<>(dossiersCollectesEnCoursLBPMoinsDeSoixanteJours);    /\*  \* RG\_CLI\_DCN\_PROCESS\_020: affichage des processus au statut « finalisé » depuis :  \* - Moins de 60 jours calendaires si l’utilisateur ne dispose pas du rôle applicatif dédié à l’accès aux pièces historisées.  \* - Moins de 2 ans si l’utilisateur dispose du rôle applicatif dédié à l’accès aux pièces historisées  \*/  **if**(!rolesService.estConseillerAvecAccesAuxPiecesHistorises(idClient)) {  /\* Filtrer les dossiers dont le statut est Finalise ou Finalise\_LBP moins de soixante jours calendaires \*/  List<DossierCollectePiecesJustificatives> dossiersCollectesFinalisesOuFinalisesLBPMoinsDeSoixanteJours = DossierCollectePredicates.*filtrerDossiersCollectes*(  dossiersCollectesFinalisesOuFinalisesLBP,  DossierCollectePredicates.*estMoinsDeSoixanteJoursCalendaires*());    /\* Liste finale: fusionner des dossiers de collectes filtrés \*/  dossiersCollectesFusionCollection.addAll(dossiersCollectesFinalisesOuFinalisesLBPMoinsDeSoixanteJours);  } **else** {  /\* Filtrer les dossiers dont le statut est Finalise ou Finalise\_LBP moins de deux ans \*/  List<DossierCollectePiecesJustificatives> dossiersCollectesFinalisesOuFinalisesLBPMoinsDeDeuxAns = DossierCollectePredicates.*filtrerDossiersCollectes*(  dossiersCollectesFinalisesOuFinalisesLBP,  DossierCollectePredicates.*estMoinsDeDeuxAns*());    /\* Liste finale: fusionner des dossiers de collectes filtrés \*/  dossiersCollectesFusionCollection.addAll(dossiersCollectesFinalisesOuFinalisesLBPMoinsDeDeuxAns);  }    /\* RG\_CLI\_DCN\_PROCESS\_020: Les processus sont affichés par date de création par ordre chronologique inverse (le premier processus de la liste est le plus récent) \*/  List<DossierCollectePiecesJustificatives> listDossiersCollectesMergeOrdreInverseDate = dossiersCollectesFusionCollection.stream()  .sorted(Comparator.*comparing*(DossierCollectePiecesJustificatives::getDateCreation).reversed())  .collect(Collectors.*toList*());    dossierCollectePiecesJustificativeList.addAll(listDossiersCollectesMergeOrdreInverseDate);    }  **catch** (HttpStatusCodeException exceptionStatusCode) {  AssemblageErrorModel model = DecodeExceptionUtils.*decode*(exceptionStatusCode);  String errorCode = model.getErrorCode();  **switch** (exceptionStatusCode.getStatusCode()) {  **case** ***NOT\_FOUND***:  String errorMessage = model.getUserMessage();  ***LOG***.warn("[listerVueDossiersCollectesPiecesJustificatives] code d'avertissement envoyé par l'assemblage : %s", errorCode);  **if** (ConstantesDossierCollecteErreurCodes.***DOSSIER\_COLLECTE\_NON\_TROUVE\_LISTER\_VUE\_DOSSIERS\_COLLECTES\_PIECES\_JUSTIFICATIVES***.equals(errorCode)) {  **throw** **new** MetierException(errorMessage, errorCode);  }  **break**;  **case** ***INTERNAL\_SERVER\_ERROR***:  ***LOG***.error("[listerVueDossiersCollectesPiecesJustificatives] code d'erreur envoyé par l'assemblage : %s", errorCode);  **throw** exceptionStatusCode;  **default**:  **throw** exceptionStatusCode;  }  }  **return** dossierCollectePiecesJustificativeList;  } |

Cette conception simplifie le test, ainsi, on test le mapper à part et le service de récupération de dossiers de collecte à part, ceci nous évitera de tomber sur le piège de confondre les tests d’intégrations et les tests unitaires.