|  |  |
| --- | --- |
| Cégep Limoilou  **Département d'informatique** | **Environnement de programmation IntelliJ IDEA** |

Table des matières

[Enregistrement d’une licence JetBrains 3](#_Toc104761907)

[Téléchargement et étapes de l’installation de IntelliJ IDEA 6](#_Toc104761908)

[Installation d’IntelliJ 8](#_Toc104761909)

[Créer un projet de programmation Java 12](#_Toc104761910)

[Créer une classe pour programmer 15](#_Toc104761911)

[Exécuter le programme 15](#_Toc104761912)

[Fichiers créés associés au projet 15](#_Toc104761913)

[Quitter IntelliJ 16](#_Toc104761914)

[Déplacer un répertoire de projet 17](#_Toc104761915)

[IntelliJ ne trouve pas le SDK (Project SDK is not defined). Que faire? 18](#_Toc104761916)

[Utilisation de l’éditeur IntelliJ 18](#_Toc104761917)

[Autre option d’installation : Installer **IntelliJ** par **JetBrains Toolbox** 19](#_Toc104761918)

[Raccourcis de l’IDE 21](#_Toc104761919)

[Raccourcis clavier 21](#_Toc104761920)

[Raccourcis pour créer du code : les **Live Templates** 21](#_Toc104761921)

[Débogage avec IntelliJ 22](#_Toc104761922)

[Activer un point d’arrêt 22](#_Toc104761923)

[Désactiver un point d’arrêt 22](#_Toc104761924)

[Lancer le débogueur 22](#_Toc104761925)

[Imprimer le code dans IntelliJ 25](#_Toc104761926)

[Afficher la structure de la classe 25](#_Toc104761927)

[Comment produire un diagramme de classes avec IntelliJ? 27](#_Toc104761928)

[Ajuster la mise en forme de votre programme 27](#_Toc104761929)

[Mettre du code en commentaires 27](#_Toc104761930)

[Enlever du code en commentaires 27](#_Toc104761931)

[Renommer une variable, une méthode, une classe 28](#_Toc104761932)

[Comment faire des tests unitaires avec **JUnit?** 28](#_Toc104761933)

[Ajouter une bibliothèque de test 28](#_Toc104761934)

[Plugin IntelliJ 28](#_Toc104761935)

[Code template 28](#_Toc104761936)

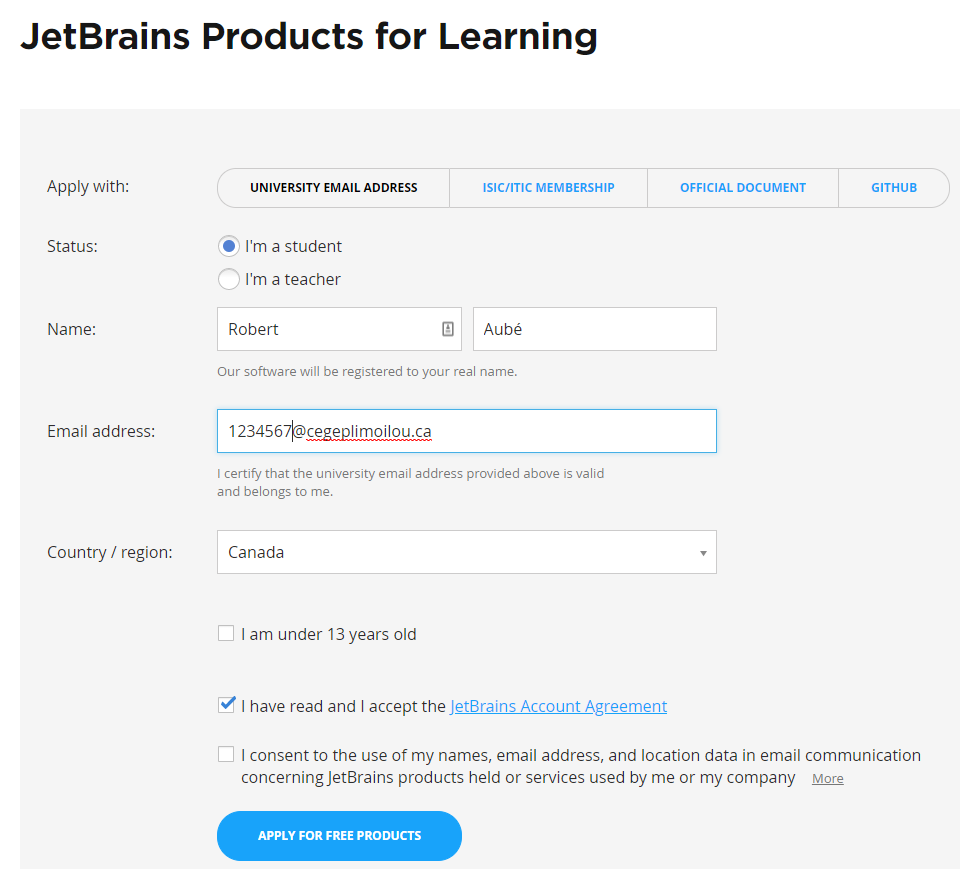
[Comment faire un exécutable avec intelliJ en utilisant launch4j? 29](#_Toc104761937)

# Enregistrement d’une licence JetBrains

L’enregistrement du logiciel **JetBrains** est optionnel. Il vous permettra d’avoir gratuitement la **version Utlimate** qui possède toutes les fonctionnalités d’IntelliJ et +. Au début de ce cours, la version communautaire répond aux besoins.

Aller sur le site : <https://www.jetbrains.com/shop/eform/students>

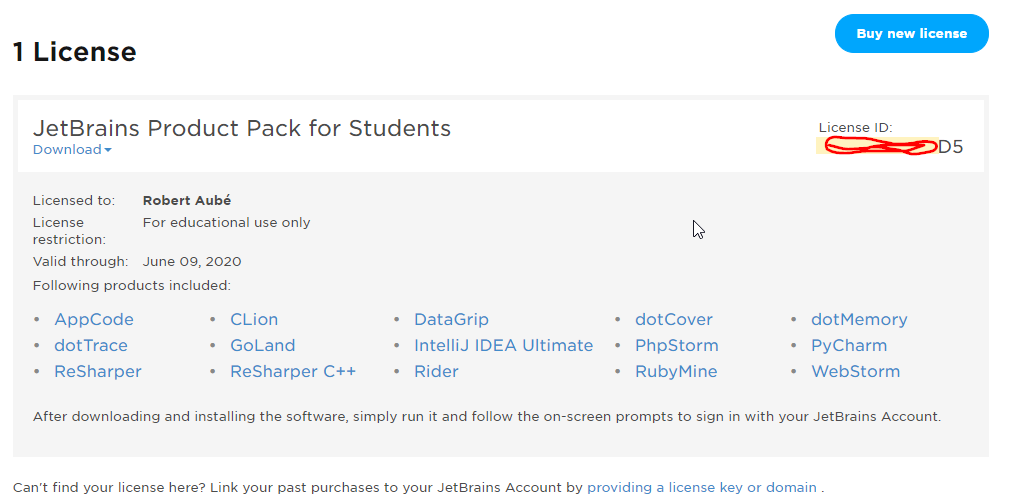
Entrez les informations requises et suivez les instructions. Notez qu’il est impératif que vous utilisiez le courriel du cégep. ([noDA@cegeplimoilou.ca](mailto:noDA@cegeplimoilou.ca))[[1]](#footnote-1)



Après avoir complété les informations, vous devriez recevoir un courriel à [noDA@cegeplimoilou.ca](mailto:noDA@cegeplimoilou.ca) pour confirmer votre compte :



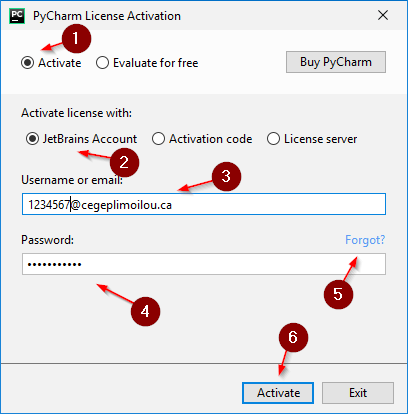
Une fois que vous aurez confirmé votre licence **JetBrains** vous aurez la possibilité d’utiliser les licences professionnelles de leurs produits. Voici un exemple d’enregistrement :



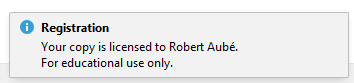
Lors de l’exécution de l’un de ces programmes nécessitant une licence, vous serez appelé à donner votre nom d’usager et mot de passe comme suit :

1. Choisir **Activate**.
2. Choisir **JetBrains Account**
3. Indiquer votre email du cégep.
4. Indiquer votre mot de passe de **JetBrains**
5. En cas d’oubli du mot de passe, il y a une procédure de récupération à suivre.
6. Une fois le tout complété correctement, appuyer sur **Activate**

L’exemple suivant présente l’activation de **PyCharm** :



Vous devriez avoir le message suivant qui s’affiche :



# Téléchargement et étapes de l’installation de IntelliJ IDEA

|  |
| --- |
| 1. Allez sur <https://www.jetbrains.com/idea/> et appuyez sur download. |
|  |
| On vous donne le choix de télécharger la version **Ultimate Commercial** ou la **Community** pour la plateforme qui vous convient.   * Si vous choisissez la **Ultimate Commercial** vous êtes obligé de vous enregistrer, car ce n’est pas une version gratuite. C’est cette version qui est utilisée au cégep. Et c’est sur cette version que seront faits tous les documents présentés dans le cours. * La version **Community** fait très bien pour le cours. Les projets sont compatibles de la version **Ultimate Commercial** à **Community** et vice versa. L’installation se fait exactement de la même façon. |
|  |
|  |

# Installation d’IntelliJ

L’installation d’un environnement de programmation doit être faite pour programmer en Java. L’opération se fait qu’une seule fois et est déjà faite sur les postes du cégep. L’installation se fait donc sur votre portable ou à la maison.

Voici une procédure d’installation :

|  |
| --- |
| 1. Exécuter l’installateur |
|  |
| 1. Aller au suivant. |
|  |
| 1. Normalement, le répertoire proposé est bon. On fait donc suivant : |
|  |
| 1. Il serait bon de cocher les cases à cocher telles que présentées ici. On fait suivant! |
|  |
| 1. On démarre l’installation. |
|  |
| 1. On peut vouloir que l’application démarre immédiatement. 2. Terminer l’installation |
|  |
| L’application IntelliJ démarre pour la première fois.   1. On pourrait avoir un fichier de configuration qui nous permettrait de récupérer des « setting » d’une autre installation d’intelliJ. Nous, ce n’est pas le cas. 2. Ok pour continuer |
|  |
| 1. Confirmez que l’on accepte la licence. |
|  |
| 1. On propose un thème d’affichage. Choisir un thème bien contrastant pour faciliter la lecture de vos programmes. 2. Sauter les autres « setting » pour utiliser ceux proposés par défaut. |
|  |
| 1. Permettre à **IntelliJ** de passer par le réseau privé. Mais ne l’autorisez pas sur des réseaux publics. |
|  |

# Créer un projet de programmation Java

|  |
| --- |
| 1. Créer un projet. Celui-ci vous permettra de placer votre programme Java. |
|  |
| 1. Il n’y a pas de librairie supplémentaire : on fait Create. |
|  |
| 1. Donnez un nom au projet. 2. Donnez l’emplacement où se trouve votre projet. Pensez que vous voudriez aussi travailler sur d’autres ordinateurs. Ainsi, il vaudrait mieux placer les fichiers du projet sur l’un des emplacements suivants :    1. Sur le lecteur U : du réseau du cégep.    2. Sur votre OneDrive du cégep : **C:\Users\1234567\OneDrive - Cégep Limoilou** où **1234567** est votre numéro de DA.    3. Sur votre clé USB 3. Appuyez sur **Create** pour passer à la création du projet. |
| 1. **Cliquez** sur le projet afin de présenter son contenu et les références. 2. On voit que le nom de projet et son emplacement sont indiqués. 3. **src** est le répertoire où on retrouvera nos classes en java. 4. On voit l’emplacement des librairies externes. C’est du code de base qu’heureusement nous n’avons pas à faire, mais que nous pouvons utiliser. Ce code nous le voulons, car nos programmes l’utiliseront. Nous y reviendrons… |
|  |
| 1. On veut créer une nouvelle classe. On appuie sur le menu contextuel (souris de droite) de **src**. 2. On choisit **New**. 3. On choisit **Java Class**. |
|  |
| 1. On choisit le **nom de la classe**. Ici son nom est **PremierPgm**. 2. Comme c’est une **classe** que l’on veut créer, conserver la valeur par défaut **Class**. Puis appuyez sur la touche **Enter**. |
|  |
| L’environnement de développement crée automatiquement la classe pour commencer à programmer en Java. |
|  |

## Créer une classe pour programmer

|  |
| --- |
| Saisir le programme suivant. Remarquer qu’une fois saisi sans erreur, des flèches vertes en marge  apparaissent pour permettre l’exécution du programme. Au contraire, s’il y a des erreurs, l’environnement de programmation souligne par différentes couleurs celles-ci.  **public class** PremierPgm {  **public static void** main(String[] args) {  System.***out***.println(**"Bonjour le monde"**);  } } |
|  |

## Exécuter le programme

|  |
| --- |
| Pour exécuter le programme, cliquez sur une flèche verte et sélectionner **Run PremierPgm.main()** |
|  |

## Fichiers créés associés au projet

|  |
| --- |
| En allant dans le répertoire où se trouve le projet, on retrouve les fichiers qui lui sont associés. Ce répertoire pourra être déplacé ou copier au besoin si l’on veut travailler sur d’autres appareils. |
|  |
| Dans le répertoire source, on retrouve le fichier **PremierPgm.java** que nous avons écrit. |
|  |

## Quitter IntelliJ

|  |
| --- |
| Pour quitter IntelliJ, faire **File/Exit** : |
|  |

# Déplacer un répertoire de projet

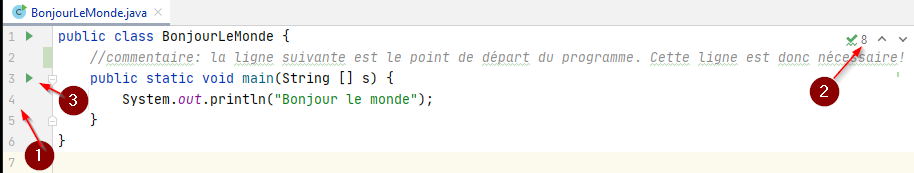
|  |
| --- |
| Il est possible de déplacer un projet d’un répertoire à un autre. Ici, on a déplacé le projet **126-projet1** dans **126 - Programmation orientée objet I**. |
|  |
| 1. On remarque ici que si on ouvre IntelliJ, le projet **126-projet1** est introuvable : (le nom du répertoire est en rouge). S’il avait trouvé, il aurait été facile de le récupérer en le sélectionnant. 2. On utilise donc Open pour récupérer notre projet **126-projet1** qui a été préalablement déplacé. |
|  |
| 1. On navigue pour retrouver le projet **126-projet1** dans **126 - Programmation orientée objet I.** 2. On sélectionne **OK**. |
|  |

## IntelliJ ne trouve pas le SDK (Project SDK is not defined). Que faire?

|  |
| --- |
| 1. Si l’on déplace un projet d’un ordinateur à un autre, il peut arriver que **IntelliJ** ne trouve plus la **SDK**. C’est le répertoire des librairies externes que notre programme java utilisera. Exemple :     Celles-ci sont essentielles. Donc, si d’un ordinateur à l’autre, les librairies ne sont pas exactement au même endroit, il ne les trouvera pas. Comme le répertoire identifie la version du SDK, une simple différence dans la version et ce problème apparaît.  Heureusement, IntelliJ nous l’indique avec une barre jaune au-dessus de la section principale du projet en indiquant **Project SDK is not defined**.   1. Dans cette situation appuyez sur **Setup SDK**. |
|  |
| 1. Dans le cadre du cours, la version du SDK n’est pas très sensible, choisissez l’une d’elles. Ici on choisit **1.8**. |
|  |

# Utilisation de l’éditeur IntelliJ

L’éditeur de programme **IntelliJ**, c’est l’endroit où vous écrivez votre code Java. Lorsque vous êtes sur votre plan de travail, vous pouvez avoir sous les yeux la quantité de vues que vous désirez. Ainsi, vous pouvez à la fois voir votre code, la hiérarchie de votre programme, la console, les messages d’erreurs, etc. Ces différentes vues forment, dans l’environnement d’IntelliJ, une « perspective ». La plupart du temps, une perspective est composée d’au moins un éditeur de programme. Voyons maintenant quelques éléments intéressants de l’éditeur :



1. Numéro de ligne du fichier texte **BonjourLeMonde.java**.
2. Le crochet vert indique que votre programme est sans erreur. Il est donc possible de l’exécuter.
3. L’icône verte permet de démarrer votre programme. Il suffit de cliquer dessus.

# Autre option d’installation : Installer **IntelliJ** par **JetBrains Toolbox**

Une autre option d’installation d’**IntelliJ** est de passer **JetBrains Toolbox**. C’est une procédure assez simple et qui a pour avantage :

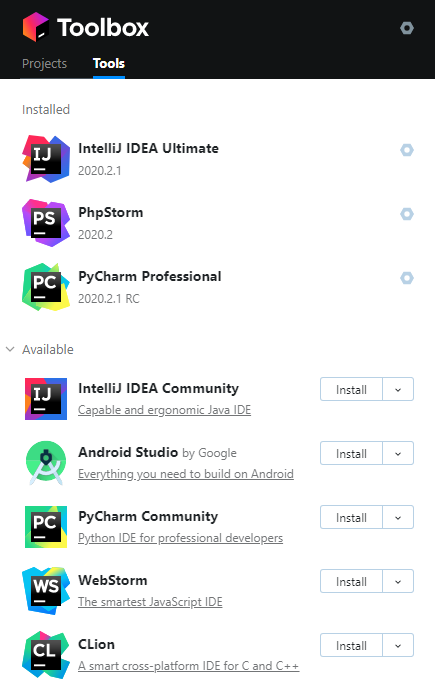
* d’avoir accès à tous vos projets à un seul endroit;
* d’avoir accès aux très bons outils de la famille **JetBrains** dont plusieurs seront utilisés au courant de la technique;
* De simplifier l’installation et la mise à jour des outils.

Pour procéder à son installation, téléchargez gratuitement **JetBrains Toolbox** à l'adresse suivante :

<https://www.jetbrains.com/fr-fr/toolbox-app/>

L'installation de cet outil est assez intuitive…

Une fois Toolbox installé, vous aurez dans la **barre de tâche** de Windows une icône  qui vous donnera accès à l’outils :



Cliquez sur **install** de l'outil **IntelliJ IDEA Community** ou **Ultimate** pour procéder facilement à l’installation.

# Raccourcis de l’IDE

## Raccourcis clavier[[2]](#footnote-2)

|  |  |
| --- | --- |
| Action | Touches |
| Reformater le code | CTRL+ALT+L |
| Afficher la structure de la classe | Alt+F7 |
| Renommer une variable, une méthode une classe | Shift+F6 |
| Afficher l’arborescence du projet | Alt+F1 |
| Mettre en commentaire une ligne ou un bloc de code avec de **//**. | Ctrl+/ |
| Mettre en commentaire une ligne ou un bloc de code avec de **/\* … \*/**. | Ctrl+Shift+/ |
| Aller à la déclaration initiale de la classe, de la variable, de la méthode sous le curseur | CTRL+B  Ou  CTRL-Clic droit |
| Afficher tous les endroits où un élément de code est utilisé dans votre projet. | Alt+F7 |
| Annuler nos modifications | CRTL+Z |
| Générer les getter et setter ou un constructeur | Alt+Inser |
| « Surround With »  « Unwrap (Inverse de surround) » | Ctrl+Alt+T  Ctrl+Maj+Suppr |

## Raccourcis pour créer du code : les **Live Templates**[[3]](#footnote-3)

Écrivez ce qu’il y a dans la première colonne puis appuyez sur la touche **tab** pour obtenir ce qu’il y a dans la 2ième colonne :

|  |  |
| --- | --- |
| Saisir le live template suivant… | Appuyez sur tab pour obtenir… |
| sout | System.*out*.println(); |
| main | public static void main(String[] args) {   } |
| iter | for (String arg : args) {   } |
| St | String |
| soutv | int i = 0; System.*out*.println("i = " + i); |
| fori | for (int i= ; i < ; i++) |

Les **Lives Templates** ont un comportement différent selon le contexte de programmation :

* Exemple 1 : main ne fonctionnera pas si vous le demandez à un endroit interdit (dans une autre méthode par exemple)
* Exemple 2 : soutv s’adaptera pour faire l’affichage de la dernière variable utilisée. Ci-haut nous avons la variable **i**.

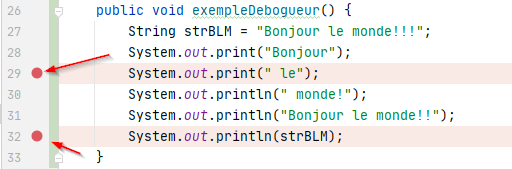
Ce n’est évidemment qu’une liste partielle des **Live Templates**. Pour obtenir la liste complète de ceux accessibles dans votre IDE, faites menu/File/Settings…/Editor/Lives Templates/Java/

# Débogage avec IntelliJ

Le débogueur permet de suivre l’évolution du code ligne par ligne. Il permet de voir les variables changer. Ainsi, grâce à lui, vous serez en mesure de trouver une erreur dans un programme et ainsi y remédier.

## Activer un point d’arrêt

Dans **IntelliJ**, on doit créer des **points d’arrêt** (breakpoint) pour indiquer où l’exécution devra s’arrêter. Pour définir un point d’arrêt, il suffit de cliquer sur la gauche de la ligne, dans la barre verticale. Lorsqu’un point d’arrêt est placé, une puce rouge  apparaît. Dans l’exemple qui suit, on peut voir 2 points d’arrêt. Ils sont à la ligne 29 et 32.

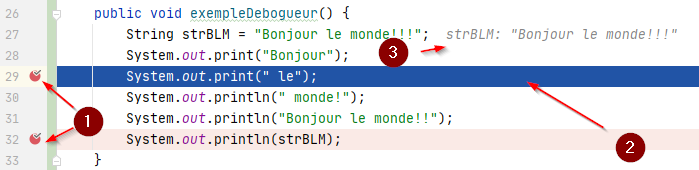


## Désactiver un point d’arrêt

Pour désactiver un point d’arrêt, il suffit de cliquer sur celui-ci.

## Lancer le débogueur

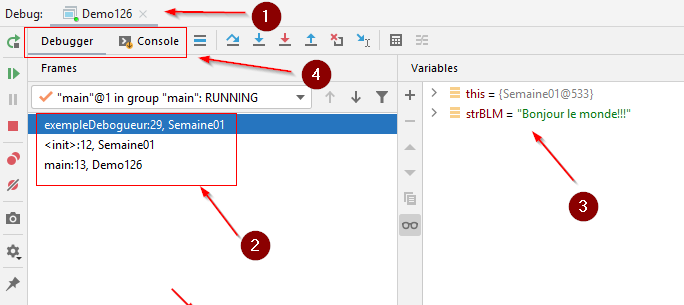
Afin de lancer le débogueur, il suffit de cliquer sur  ou appuyer sur Maj-F9.



Dans la capture présentée ci-haut, on voit…

* En 1 que le débuggeur est en cours d’exécution. Les puces rouges , on maintenant un petit crochet .
* En 2, c’est la ligne où le débuggeur s’est arrêté en raison du point d’arrêt. Le programme est donc rendu à la **ligne 29**. Cette ligne est surlignée en **bleu**.
* En 3, **IntelliJ** présente la valeur des variables en grisé pâle. On sait donc que lorsque le programme est rendu d’exécuter la ligne 29 la valeur de strBLM est "Bonjour le monde !!!".

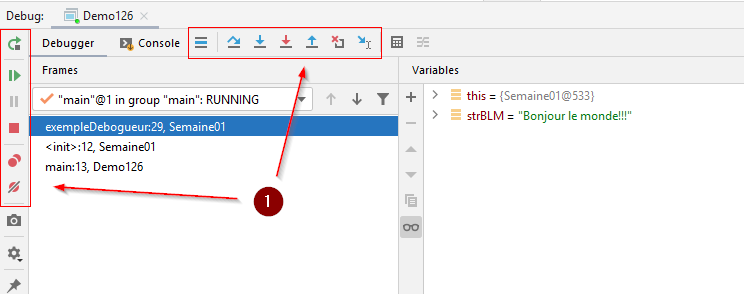
L’action de lancer le débogueur ouvre également la perspective de **Debug**.



Dans la capture présentée ci-haut, on voit…

* En 1, on voit bien que c’est le programme **Demo126** qui a été démarré dans le programme.
* En 2, c’est la suite des appels qui a permis de se rendre à la ligne 29 de exempleDebogueur : D’abord de la ligne 13 du main, on s’est rendu à la ligne 12 de Semaine01 qui a enfin appelé la méthode exempleDebogueur. Le débogueur est rendu à la ligne 29 de exempleDebogueur.
* En 3, le débogueur présente la valeur des variables en grisé pâle. On sait donc que lorsque le programme est rendu à exécuter la ligne 29 et la valeur de strBLM est "Bonjour le monde !!!".
* En 4, **IntelliJ** vous permet de passer de l’affichage en **console** au **débogueur**.

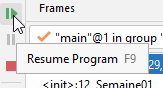
Pour poursuivre l’exécution du programme et voir l’évolution des états du programme le contrôle se fait à partir des boutons de la perspective du débuggeur :



Nous avons les possibilités suivantes[[4]](#footnote-4) :

|  |  |  |  |
| --- | --- | --- | --- |
| Action | icône | Raccourci | Explications |
| **Resume** |  | F9 | Continue l’exécution du programme depuis l'instruction où il a été suspendu jusqu’au prochain point d’arrêt ou jusqu’à la fin du programme. Dans l’exemple, si on appuie sur F9, le programme s’exécutera sans arrêt jusqu’à la ligne 32 où se trouve le prochain point d’arrêt. |
| **Stop** |  | CTRL-F2 | Termine l’exécution du programme à l’endroit où il est rendu. |
| **Step into** |  | F7 | Exécute pas à pas les instructions du programme. S’il y a un appel de méthode, entre dans celle-ci pour exécuter pas à pas ses instructions. |
| **Step over** |  | F8 | Exécute chacune des lignes du programme, s’il y a un appel de méthode, la méthode est exécutée, mais sans entrer pas à pas dans celle-ci. |
| **Step out** |  | Maj-F8 | Quitter l’exécution pas-à-pas d’une méthode en cours d’exécution et de revenir à l’appelant. |
| **Rerun demo** |  | Ctrl-F5 | Terminer le programme et réexécuter en mode débogage. |
| **Run to Cursor** |  | Alt-F9 | Continue l’exécution du programme depuis l'instruction où il a été suspendu jusqu’à l’endroit où vous avez placé le curseur dans le programme et cela sans même avoir placé un point d’arrêt.  Il s’arrêtera quand même au point d’arrêt et complétera l’exécution jusqu’à la fin du programme s’il ne passe pas par l’endroit dicté. |
| **Mute Breaks Points** |  |  | Ne plus tenir compte des breaks points. Ils se désactivent temporairement. Les puces rouges  sont maintenant grisées . |
| **View Breaks Points…** |  | CTRL-SHIFT-F8 | Ouvre une fenêtre qui permettra de voir rapidement où se trouvent les Breaks Points dans le programme exécuté. |

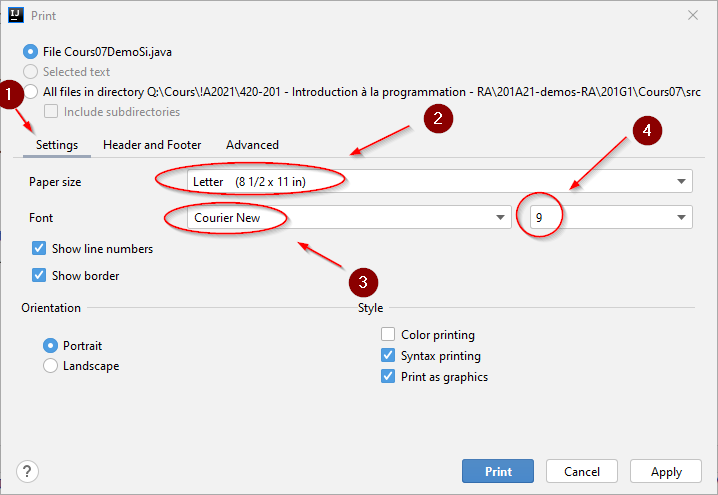
En passant, la souris et en l’arrêtant sur une icône, **Intellij** nous donne l’explication de celle-ci et la touche de raccourci à utiliser. Ici, n voit qu’en passant sur l’icone , l’Action est « resume » et la touche de raccourci est F9.



# Imprimer le code dans IntelliJ

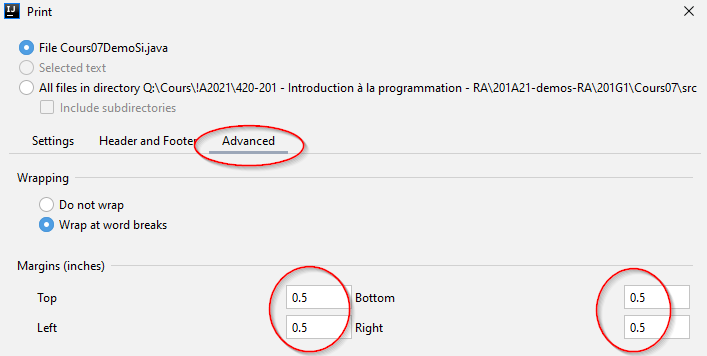
Attention : au cégep et probablement chez vous, les paramètres (setting) par défaut pour imprimer directement d’IntelliJ ne sont pas bons. L’impression présentera votre code, mais le formatage ne sera pas simple à lire.

Pour accéder ces settings, vous vous rendez directement dans le menu File/Print…



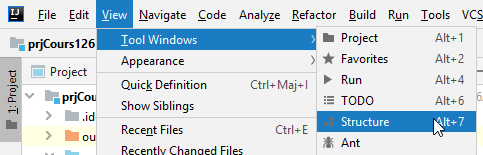
Un bon setting (1) est Letter (2), Courier New (3), 9 (4).

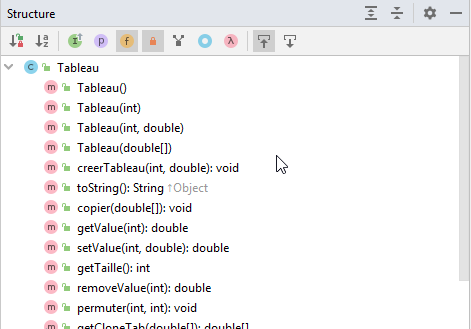
Et mettre toutes les marges à 0.5 pouces…



# Afficher la structure de la classe

On peut naviguer rapidement à l’intérieur d’une classe en utilisant la fenêtre **Structure** :





# Comment produire un diagramme de classes avec IntelliJ?

1. Enregistrer votre diagramme de classe des classes de **dataCenter** sous forme d’image :

(Souris de droite sur package **dataCenter**, *Diagrams/Show Diagram…*)

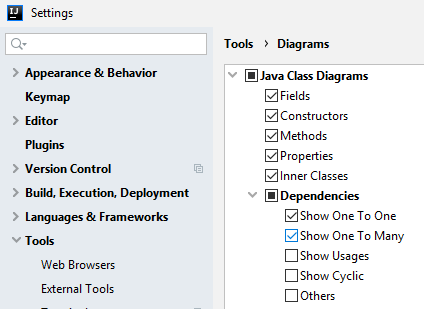
Pour être en mesure de faire des **diagrammes de classe,** il est prérequis d’avoir la version**IntelliJ IDEA Ultimate[[5]](#footnote-5)**. Le plugin **Simple UML** doit être présent et activé (File/Settings…/Plugins/Install).

****

Ainsi, **IntelliJ IDEA Ultimate** vous permet de générer un **diagramme de classes** sur un package de votre projet. Ces diagrammes reflètent la structure des classes et des méthodes réelles de votre application.

Pour afficher le diagramme sur un package :

1. Dans la fenêtre de l'outil **Project**, cliquez avec le bouton droit sur un **package** pour lequel vous souhaitez créer un diagramme et sélectionnez **Diagrams/Show Diagram (Ctrl+Alt+Shift+U )**.
2. Dans la liste qui s'ouvre, sélectionnez **Java Class Diagram**. **IntelliJ IDEA** génère un diagramme UML pour les classes et leurs dépendances.



# Ajuster la mise en forme de votre programme

Pour arriver à bien comprendre un programme, le formatage du code et son indentation sont essentiels afin d’améliorer la relecture et le débogage. Ainsi, pour éviter de faire cette mise en forme manuellement, faites **CTRL+ALT+L**.

# Mettre du code en commentaires

Il arrive souvent que l’on désire placer une partie du code en commentaires pour tester des parties isolées de programme. Au lieu de le faire manuellement, on utilise CTRL + / pour mettre en commentaires la sélection.

## Enlever du code en commentaires

Ce qui s’ajoute doit pouvoir s’enlever par la suite. Il suffit d’appuyer encore sur CTRL + / pour retirer les commentaires.

# Renommer une variable, une méthode, une classe

Lorsque l’on veut renommer une variable, une méthode ou une classe, cela peut devenir fastidieux si nous devons faire les modifications partout où il y a des répercussions. La commande **SHIFT+F6** permet de faire cette opération automatiquement.

# Comment faire des tests unitaires avec **JUnit?**

Ref :<https://www.jetbrains.com/help/idea/testing.html>

IntelliJ IDEA fonctionne avec plusieurs frameworks de test prêts à l'emploi, par exemple JUnit, TestNG , Cucumber ou Arquillian JUnit .

## Ajouter une bibliothèque de test

Suivez ces étapes pour ajouter une bibliothèque de test dans IntelliJ IDEA :

Dans le menu principal, sélectionnez Fichier | Structure du projet (**Ctrl+Alt+Shift+S**) ou cliquez le bouton Structure du projet sur dans la barre d'outils.

Sous Paramètres du projet, sélectionnez Bibliothèques et cliquez sur le bouton Nouvelle bibliothèque de projets | De Maven .

Dans la boîte de dialogue qui s'ouvre, spécifiez l'artefact de bibliothèque nécessaire, par exemple : org.junit.jupiter:junit-jupiter:5.4.2ou org.testng:testng:6.14.3.

Appliquez les modifications et fermez la boîte de dialogue.

## Exécuter tous les tests dans IntelliJ

Pour exécuter tous les tests dans un dossier, sélectionnez ce dossier dans la fenêtre de l'outil Projet **Ctrl+Shift+F10** et appuyez sur ou sélectionnez Exécuter les tests dans 'dossier' dans le menu contextuel

# Plugin IntelliJ

Extensions IntelliJ:

* Visual Debugger
* Code comment -
* Jump To Line - permet d'accéder à n'importe quelle ligne de code pendant le débogage
* Graphical Navigator
* Regex Tester
* Learn and teach

# Code template

1. Modifier le **code template** d’intelliJ des **classes**, des **interfaces** et des **enum** avec :

*File / Settings / File and Code Templates*

Y ajouter le code suivant dans le haut des fichiers pour bien identifier tous vos fichiers :

// 420-201-H22 – TP5 – POO ($DATE)

// Par : Aubé, Robert (1234567) & Fortin Denis (7654321)

# Comment faire un exécutable avec intelliJ en utilisant launch4j?

D’abord :

<https://www.jetbrains.com/idea/guide/tutorials/hello-world/packaging-the-application/>

Ensuite utiliser **launch4j :**

<https://www.youtube.com/watch?v=MyMPPuYGN-U>

1. Vous pouvez récupérer vos courriels de l’adresse en suivant ce lien (<https://outlook.office.com/mail/>) [↑](#footnote-ref-1)
2. Références :

   <https://blog.sodifrance.fr/astuces-et-raccourcis-intellij/>

   <https://blog.jetbrains.com/fr/2020/03/09/les-15-meilleurs-raccourcis-dintellij-idea/> [↑](#footnote-ref-2)
3. Référence :

   <https://blog.jetbrains.com/fr/2020/06/10/ecrivez-du-code-plus-rapidement-grace-aux-live-templates/> [↑](#footnote-ref-3)
4. Référence :

   <https://www.jetbrains.com/help/idea/stepping-through-the-program.html> [↑](#footnote-ref-4)
5. Référence :

   <https://www.jetbrains.com/help/idea/class-diagram.html#analyze_class> [↑](#footnote-ref-5)