Aris Xanthos
Unil, SLI

Expressions régulières pour l'extraction de données semi-structurées

I. Objectifs

- importer un texte dans Textable à partir d'une source en ligne
- découvrir la syntaxe de base des expressions régulières (regex)
- utiliser les regex pour extraires des informations à partir de données textuelles semi-structurées
- exporter des données textuelles dans un format permettant leur importation dans un autre logiciel (en l'occurrence Iramuteq)

II. Prérequis

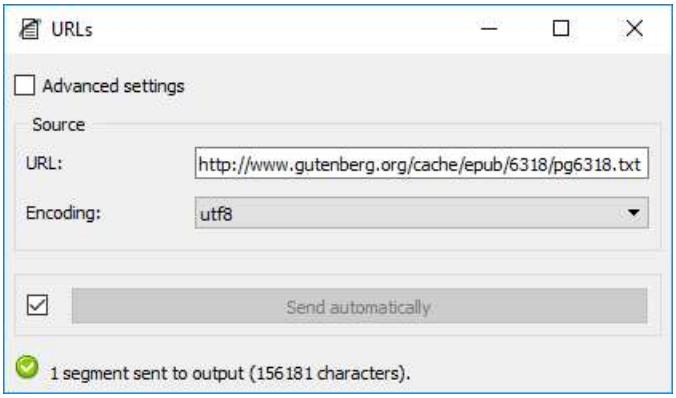
Les étapes suivantes doivent être effectuées au préalable:

- Installer Orange Canvas 3.7 (librement disponible sur https://orange.biolab.si/download/)
- Installer l'extension Textable (libre, voir instructions sur http://textable.io/get-started/)
- Se familiariser avec les bases de la programmation visuelle avec Textable, p.ex. en suivant les étapes du TP *Import de données XML et création de matrices documents–termes*

III. Importer des données dans Orange Canvas à partir d'une URL

1. Exécutez *Orange Canvas*.

2. Créez une copie du widget **Textable > URLs** et configurez-la pour importer le texte de *l'Avare* de Molière accessible sur le site du projet Gutenberg à l'URL suivante (en utf8): http://www.gutenberg.org/cache/epub/6318/pg6318.txt:

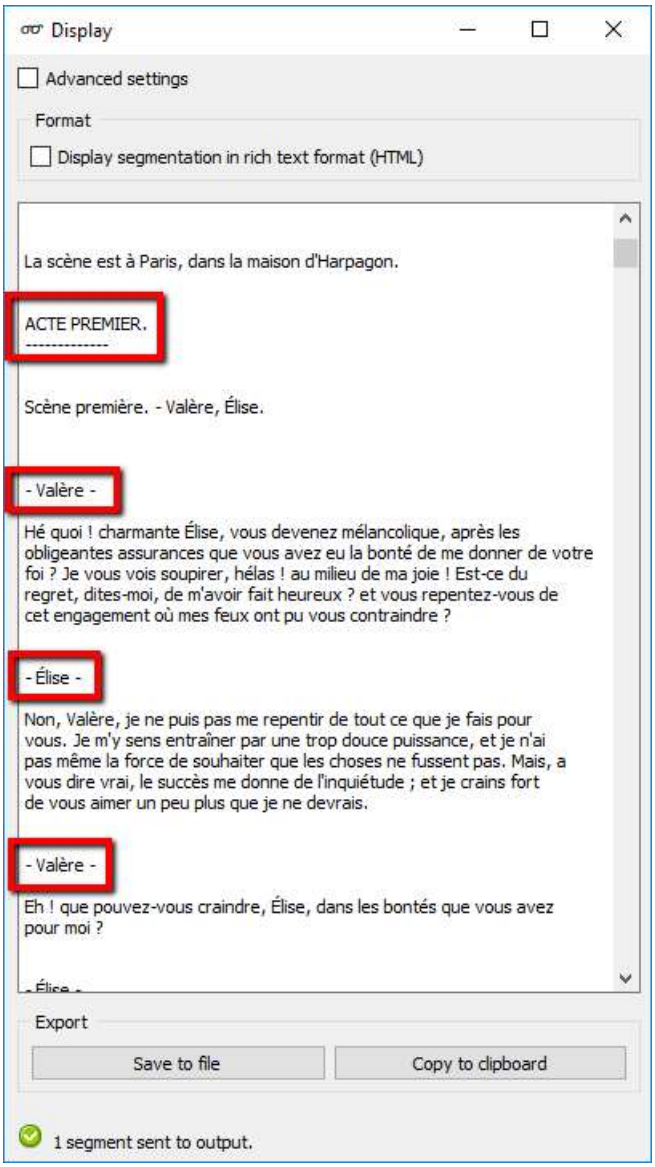


IV. Première approche de segmentation par regex

1. Connectez **URLs** à une nouvelle instance de **Textable > Display** et prenez un instant pour examiner la structure du texte (voir figure page suivante).

   Cet examen montre que bien que les données ne soient pas formellement structurées au sens où peut l'être un document XML par exemple, elles manifestent des régularités formelles suffisantes pour permettre d'identifier plusieurs niveaux d'unités textuelles:

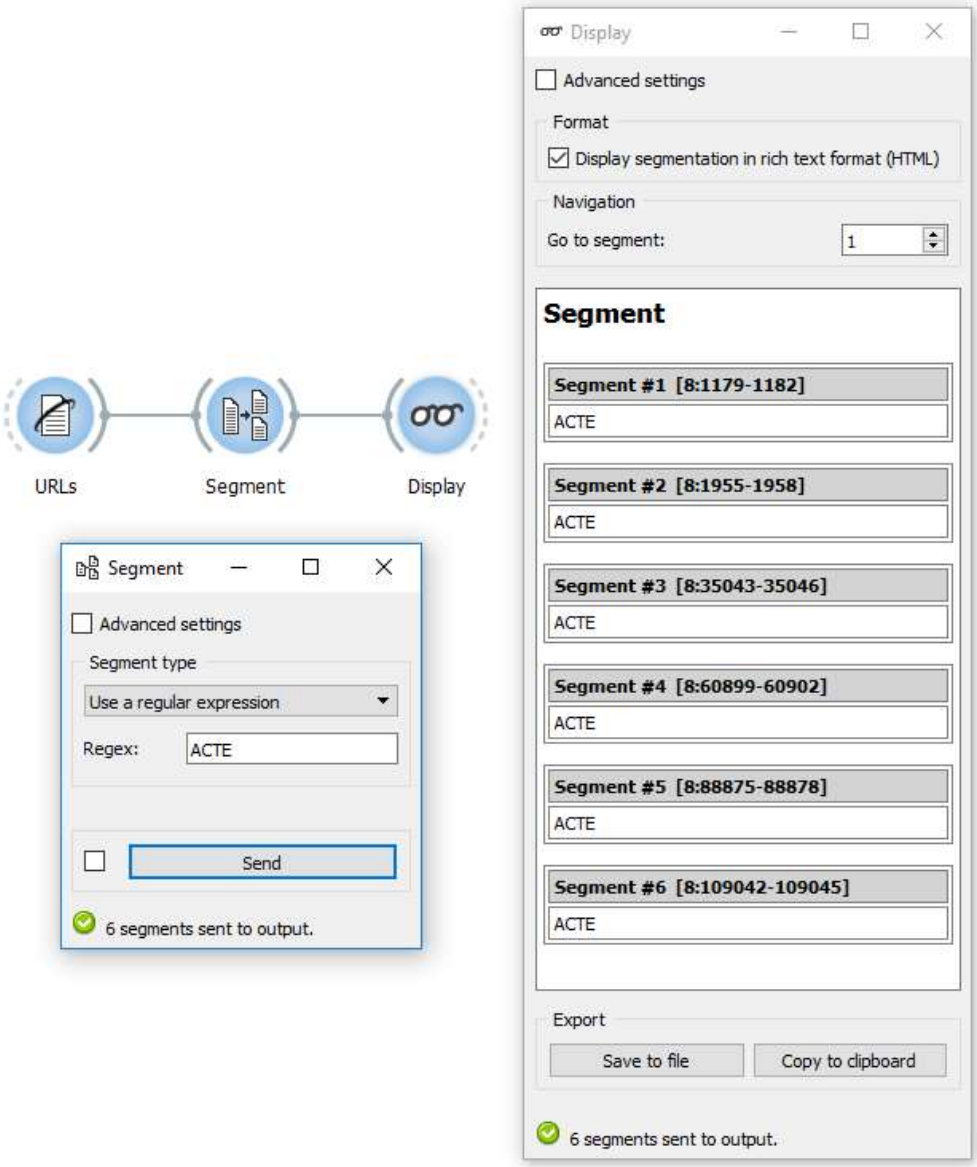
   - les actes commencent toujours par la chaîne *ACTE*
   - les répliques commencent toujours par *tiret + espace + nom du personnage + espace + tiret*



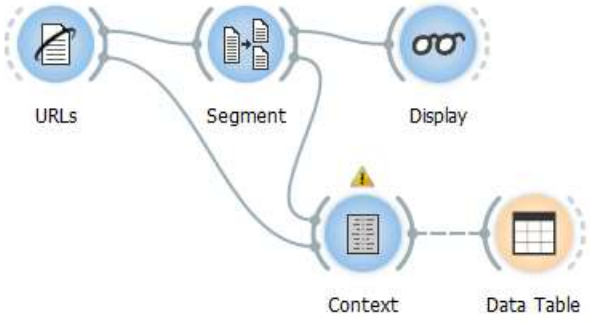
2. Dans ce TP, nous apprendrons à utiliser la syntaxe des regex[1] pour extraire et exploiter ce type d'informations semi-structurées. Commencez par insérer entre **URLs** et **Display** une nouvelle copie de **Textable > Segment**:

3. La fonction de **Segment** est de repérer les segments du texte correspondant à une regex donnée et de les utiliser pour créer une nouvelle segmentation. Pour tester ce principe, saisissez la chaine *ACTE* (en majuscules) dans le champ **Regex** de **Segment**:

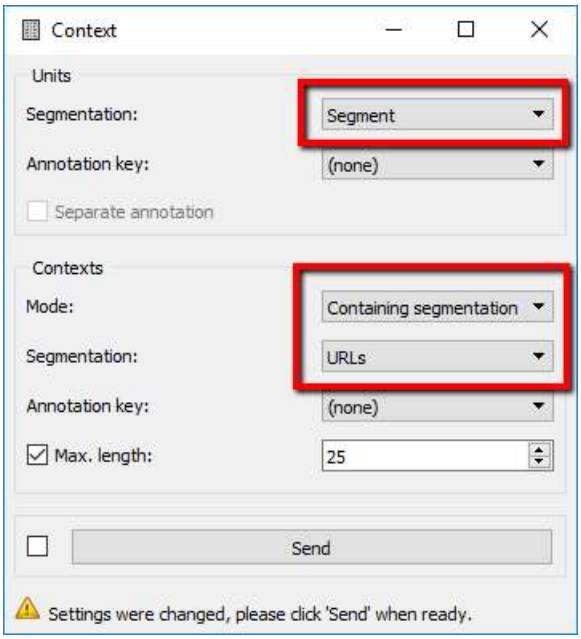
[1] Une recherche sur le web vous conduira à des milliers de ressources en ligne permettant de vous familiariser avec cette syntaxe (p.ex. https://regexone.com/, https://regexr.com/, https://www.debuggex.com/, https://regexcrossword.com/).



4. Nous apprenons ainsi que cette chaine apparait 6 fois dans le document. Cette requête ne semble pas assez précise puisqu'il s'agit d'une pièce en 5 actes. Pour en savoir plus sur le contexte des occurrences de *ACTE*, connectez **URLS** *et* **Segment** à une nouvelle copie de **Textable > Context**, et connectez à son tour **Context** à une nouvelle copie de **Data > Data Table**:



5. Configurez **Context** comme sur la figure ci-dessous pour indiquer que les éléments dont on veut examiner le contexte sont ceux découpés par **Segment**, et que le contexte en question est donné par le texte importé avec **URL**:



6. Après avoir cliqué sur **Send**, un apercu du contexte des occurrences de la chaine *ACTE* est visible dans **Data Table**, et permet de déterminer que l'occurrence "en trop" est celle contenue dans le mot *ACTEURS*:

	__left__	segm	__right__
1		ACTE	URS
2	la maison d'Harpagon.	ACTE	PREMIER.----------
3	estique de la sorte !	ACTE	SECOND.----------
4	rer bonne récompense.	ACTE	TROISIÈME.----------
5	envie de me ruiner ?	ACTE	QUATRIÈME.----------
6	ndraimoi-même après.	ACTE	CINQUIÈME.----------

En effet, en ajoutant un espace après *ACTE* dans l'interface de **Segment**, on restreint l'ensemble des segments identifiés à ceux qui marquent le début d'un acte.

V. Découper le texte en segments commençant par une chaîne donnée

1. En pratique, nous ne sommes pas simplement intéressés à connaître le nombre d'occurrences de la chaine *ACTE*,[2] mais bien plutôt le *contenu* de chaque acte, par exemple afin de pouvoir le traiter comme un contexte/document dans la construction d'une matrice documents–termes.

   Commençons donc notre exploration des outils que fournit la syntaxe des regex par le symbole "." (*point*), qui représente n'importe quel caractère à l'exception du retour à la ligne.

[2] Il n'y a pas besoin pour cela d'autre chose qu'une requête "full-text".

Saisissez donc la chaine *ACTE .* (*ACTE + espace + point*) dans **Segment** et observez le résultat dans **Display**:

Segment #1 [8:1955-1960]
ACTE P

Segment #2 [8:35043-35048]
ACTE S

Segment #3 [8:60899-60904]
ACTE T

Segment #4 [8:88875-88880]
ACTE Q

Segment #5 [8:109042-109047]
ACTE C

Comme on le voit, la notation "." est l'une de celle qui permet d'identifier des segments dont contenu textuel est variable.

2. En poursuivant sur cette lignée, ajoutez le signe *+* (*plus*) immédiatement après "." dans la configuration de **Segment** (la requête est donc "*ACTE .+*", *ACTE + espace + point + plus*):

Segment #1 [8:1955-1967]
ACTE PREMIER.

Segment #2 [8:35043-35054]
ACTE SECOND.

Segment #3 [8:60899-60913]
ACTE TROISIÈME.

Segment #4 [8:88875-88889]
ACTE QUATRIÈME.

Segment #5 [8:109042-109056]
ACTE CINQUIÈME.

Le symbole *+* est un *quantifieur* qui signifie "1 ou plusieurs fois" ce qui le précède immédiatement (ici "."); le résultat est que la regex renvoie toutes les occurrences de *ACTE* suivies par un espace puis 1 ou plusieurs caractères quelconques à l'exception du retour à la ligne.

Notez que le comportement par défaut du moteur de recherche est de renvoyer les plus longues chaines possibles qui satisfont la regex, et donc ici de capturer toute la fin de la ligne à partir de *ACTE + espace*.

3. Il est possible de franchir la limite de la fin de la ligne en spécifiant que le signe "." signifie n'importe quel caractère *y compris le retour à la ligne*. Ajoutez pour cela le code *(?s)* à la fin

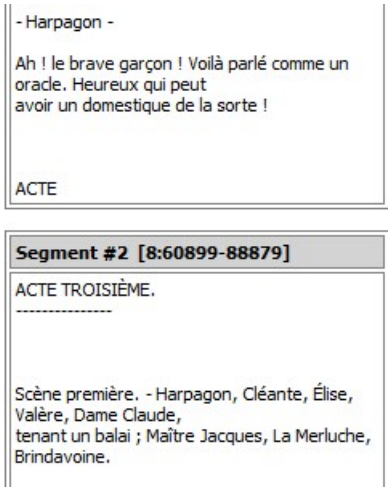
de la regex, qui devient donc "*ACTE .+(?s)*". Ce faisant, vous constaterez que la requête ne renvoie plus qu'un seul segment, constitué de l'ensemble du texte depuis la première occurrence de *ACTE + espace.* C'est en effet le résultat du comportement par défaut du quantifieur *+,* qui consiste à capture la plus longue chaine possible.

4. Pour remédier à cet excès de gourmandise, il faut modifier notre requête de deux façons:

    a. le comportement des quantifieurs comme *+* peut être modifié pour renvoyer une chaine *aussi courte que possible* en les faisant suivre par le signe *"?"*

    b. pour s'assurer toutefois que la chaine renvoyée s'étend jusqu'à la fin de chaque acte, on peut indiquer dans la requête qu'elle doit se terminer par une occurrence de *ACTE + espace*

    La requête ainsi modifiée s'écrit donc *"ACTE .+?ACTE (?s)"*. Hélas! Cette approche renvoie plus d'un segment, mais pas plus de deux; en effet, comme le premier segment identifié se termine par la chaine *ACTE + espace*, c'est par la troisième occurrence que commence le second segment:



5. Il existe bien sûr une solution pour sortir de cette impasse: indiquer que chaque segment doit *être suivi par ACTE + espace* (par opposition à *se terminer par ACTE + espace*). La notation pour cela est assez compliquée: *"ACTE .+?(?=ACTE)(?s)"*, où *(?=X)* signifie donc "suivi par *X*".

6. Avec cette dernière modification, nous sommes presque au bout de nos peines. Il ne manque plus qu'un acte: le dernier, qui (il faut bien l'admettre) n'est *pas* suivi par *ACTE + espace*. Ce qui signale (de façon unique) le dernier acte est en fait une longue série de tirets (au moins 50, à vue de nez), si bien qu'il faut en fait indiquer que nos segments doivent être suivis par *ACTE + espace* OU par au moins 50 tirets. OU s'écrit ici au moyen de la barre verticale "|" (*alt + ctrl + 7* sur PC, *alt + 7* sur Mac), et "au moins 50 répétitions de ce qui précède" s'écrit *"{50,}"* (c'est un autre exemple de quantifieur). La requête définitive est donc :

ACTE .+?(?=ACTE |-{50,})(?s)

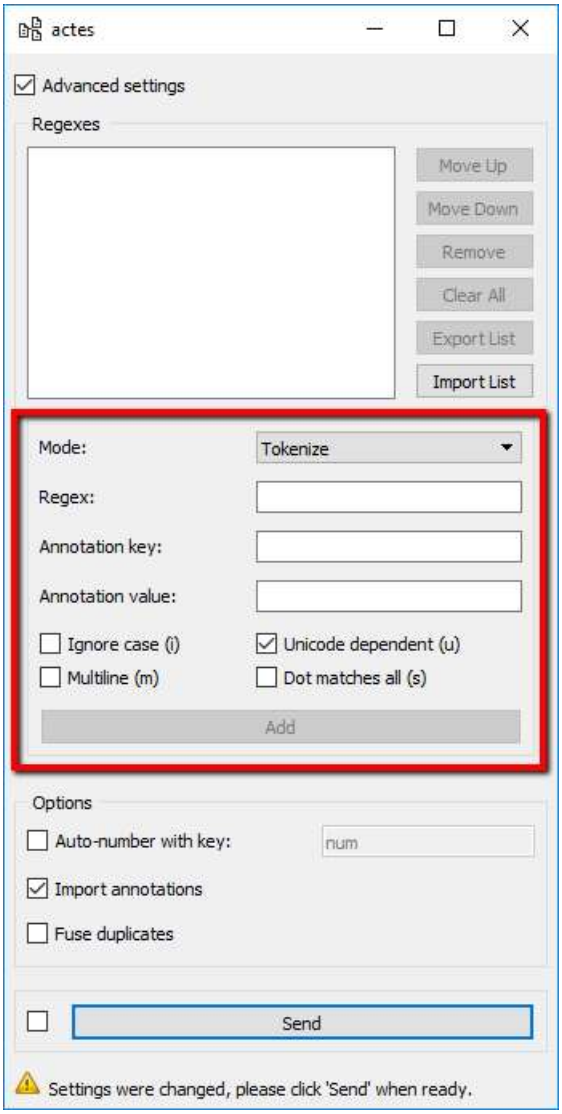
Exercice 1

1. Renommez **Segment** en a*ctes*, connectez-le à une nouvelle copie de **Segment**, et renommez celle-ci *repliques*. Prenez un instant pour essayer de trouver une regex permettant d'identifier les 967 en-têtes de répliques contenues dans les 5 actes (voir IV.1 ci-dessus).

2. Au moyen des widgets **Count** et **Data Table**, établissez la fréquence de chaque type d'en-tête dans l'ensemble du texte.

VI. Créer des annotations avec une regex

1. Le widget **Segment** permet d'associer des annotations aux segments qu'il identifie. Dans ce qui suit, nous exploiterons cette possibilité pour annoter chaque acte au moyen du numéral qui le spécifie (premier, second, etc.). Nous utiliserons pour cela les réglages avancés (**Advanced settings**) du widget:



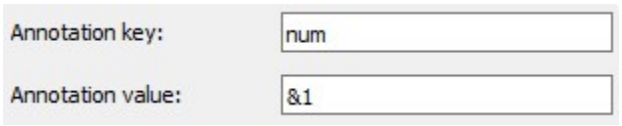
Dans ce mode, le widget peut stocker puis appliquer successivement plusieurs regex. Chacune doit pour cela être configurée d'abord dans la section centrale de l'interface (voir figure ci-dessus), puis ajoutée à la liste **Regexes** en cliquant sur **Add**.

2. Pour nos objectifs, une seule regex suffit. Copiez la regex définie au point V.6 ci-dessus, collez-la dans le champ **Regex** des réglages avancés et modifiez-la comme suit:

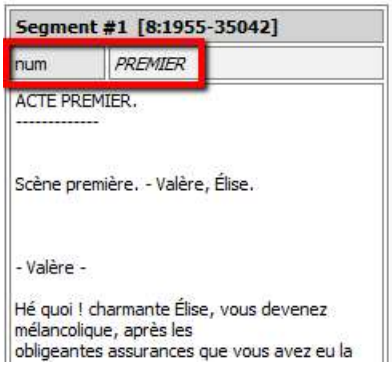
   ACTE (PREMIER|SECOND|TROISIÈME|QUATRIÈME|CINQUIÈME).+?(?=ACTE |-{50,})(?s)

   Le sens de l'expression entre parenthèses ajoutée ici devrait être clair à ce stade: elle spécifie que l'un des 5 adjectifs numéraux spécifiés doit apparaître à la suite de *ACTE + espace.*

3. Les parenthèses qui encadrent les numéraux dans l'expression ci-dessus jouent un double rôle: d'une part elles délimitent la portée de la barre verticale, d'autre part elles "capturent" la portion de la chaine correspondante dans les segments identifiés et permettent de l'utiliser pour créer une annotation. A cet effet configurez les champs **Annotation key** et **Annotation value** comme suit:



   L'effet de ces paramètres est de créer, pour chaque acte, une annotation *num* dont la valeur est la sous-chaine identifiée par la première paire de parenthèses dans la regex, soit ici *PREMIER*, *SECOND*, etc. Validez avec **Add**, cliquez sur **Send** si nécessaire et vérifiez le résultat avec **Display**:



Exercice 2

En vous appuyant sur les étapes effectuées dans l'exercice 1 ci-dessus, créez une matrice document–termes dont les lignes sont les actes (représentés par l'adjectif numéral correspondant) et les colonnes sont les types d'en-têtes de répliques:

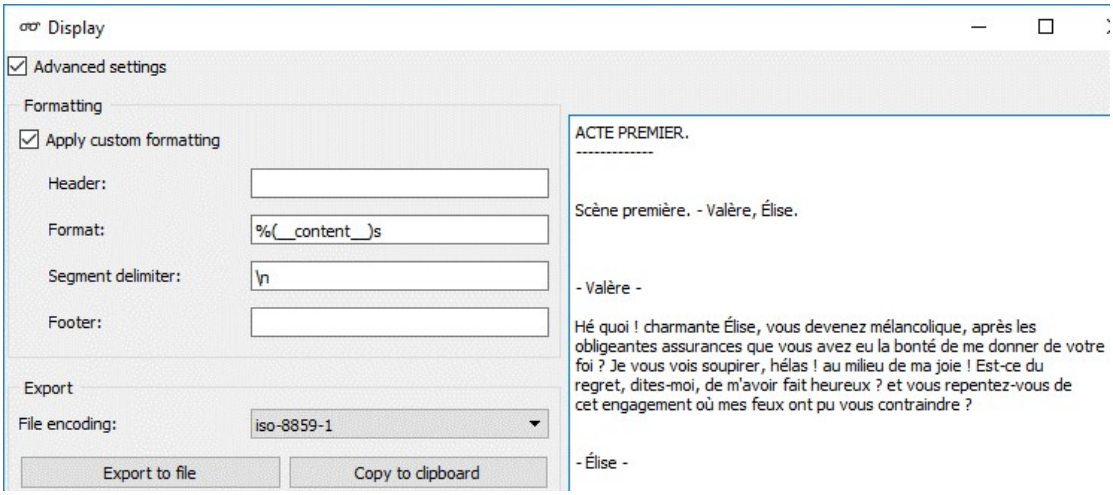
context	- Valère -	- Élise -	- Cléante -	- Harpagon -
PREMIER	31	44	39	108
SECOND	0	0	27	45
TROISIÈME	25	2	26	74
QUATRIÈME	0	3	65	53
CINQUIÈME	46	2	4	80

VII. Exportation de texte reformaté

1. Pour terminer, nous illustrerons la possibilité d'exporter les informations extraites au moyen des regex dans un format permettant leur exploitation ultérieure dans un autre logiciel, en

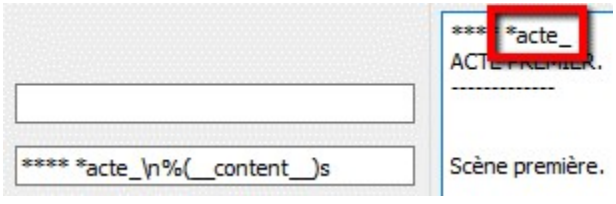
l'occurrence Iramuteq. Celui-ci requiert que chaque "document" du corpus soit précédé par une ligne commençant par 4 astérisques, qui peuvent être suivies par 1 ou plusieurs "variables étoilées", c'est-à-dire des paires attribut-valeur respectant le format **attr_val*.

2. Dans notre cas, les documents sont les actes et ils sont parés d'un unique attribut *num*. Pour générer le formatage requis, ouvrez le widget **Display** connecté à la sortie du widget renommé *actes* et activez les réglages avancés (**Advanced settings**):



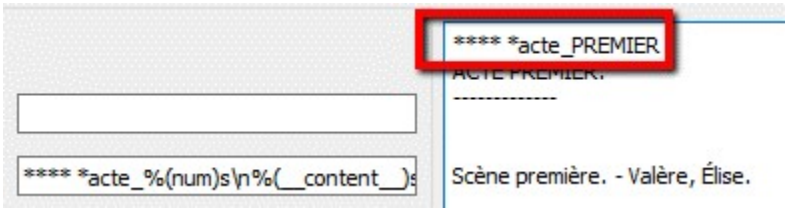
3. Sélectionner l'option **Apply custom formatting** vous permet définir vous-même le format d'affichage de chaque segment, par le biais du paramètre **Format**. Par défaut, il s'agit simplement du contenu du segment – c'est ce que signifie le code prédéfini *%(__content__)s.* Insérez au début de ce champ la chaine *****\n* pour faire précéder chaque acte par la ligne aux 4 astérisques requise par Iramuteq (*"\n"* signifiant "retour à la ligne"):

4. Ajouter les "variables étoilées" n'est pas beaucoup plus compliqué. Commencez par insérer la partie invariable de cette notation (**acte_*) en l'insérant à l'endroit voulu sous **Format** (soit à la suite des 4 astérisques, après un espace):



5. Pour ajouter la partie variable (*PREMIER*, *SECOND*, etc.), il suffit d'insérer après l'underscore une référence à l'annotation correspondante (*num*), en respectant le format suivant:

%(num)s



6. Le fichier ainsi recodé peut être exporté en cliquant sur **Export to file** après avoir sélectionné l'encodage approprié (p.ex. *utf8*), pour être ensuite importé dans Iramuteq.

Exercice 3

Reprenez le corpus XML utilisé dans le TP *Import de données XML et création de matrices documents–termes* et tâchez d'en créer une version reformatée pour l'import dans Iramuteq (chaque discours inaugural correspondant à un document annoté par deux "variables étoilées", l'année du discours et le parti du président).