

**Web services et annotations**  
**Projet « scripting pour l'agrégation automatique d'annotations »**

## 1. Contexte scientifique : annotation fonctionnelle de listes de gènes.


Les traitements bioinformatiques et biostatistiques des données en sciences omiques (notamment le cas d'expériences à haut débit à visée d'analyse différentielle) fournissent en sortie de longues listes de gènes correspondant à plusieurs conditions d'études ou phénotypes. Par ailleurs, les biologistes ont parfois besoin de constituer des listes de gènes, point de départ d'une expérience ou bien pour synthétiser une recherche bibliographique etc.

Dans les deux cas, **ces listes de gènes doivent encore être annotées fonctionnellement** : soit pour permettre aux biologistes une interprétation biologique des processus cellulaires en cause dans le mécanisme étudié (une seconde étape d'analyse d'enrichissement fonctionnelle sera nécessaire), soit pour constituer un fichier de référence informatif fonctionnellement et facile d'utilisation.

Le processus est fastidieux à la main, notamment par la dispersion des ressources et le travail répétitif : banques de séquences, banques de domaines, de terminologies (Gene Ontology, Disease Ontology), de voies métaboliques, de familles de protéines, de structure 3D etc. Pour de longues listes de gènes, l'agrégation d'annotations à la main devient impossible.

## 2. Outils

- ✓ **Des outils conviviaux**, ne nécessitant pas de programmation ont été développés pour permettre l'assignation automatique d'annotations aux gènes ; l'outil **Biomart d'Ensembl** est un très bon exemple : « Export custom datasets from Ensembl with this data-mining tool ». Cependant, une limitation est le nombre de type d'annotations que l'on peut agréger.

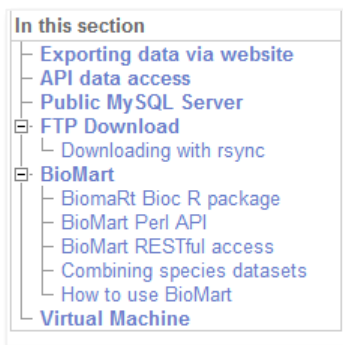
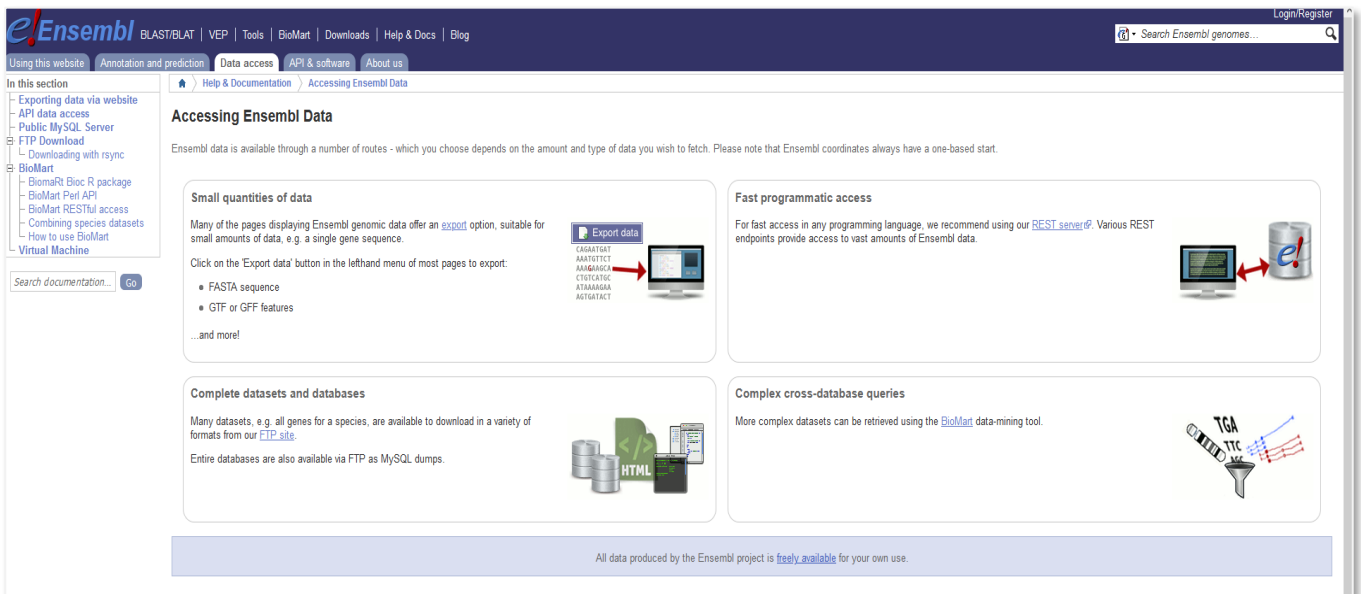
 [-An introduction to BioMart](#) : it can be used to export data from Ensembl, including information such as tables of gene IDs, gene positions, associated variations, and protein domains, or sequences. This video begins at [www.ensembl.org](http://www.ensembl.org) and walks you through the BioMart interface to show you how to obtain information for multiple genes, and for a region of the chromosome.

- ✓ **Des outils pour un accès programmatique** aux banques de données, ont été développés : **API ou Application Programming Interface** (interface de programmation applicative ou interface de programmation d'application). Il s'agit d'un ensemble de fonctions qui facilitent, via un langage de programmation, l'accès aux services d'une application (Web services). Il existe plusieurs protocoles de communications utilisés par les API, parmi lesquels le protocole **Representational State Transfer (REST)**, le plus utilisé aujourd'hui en bioinformatique.

- Exemples :

- Ensembl : API Rest

- <http://www.ensembl.org/info/data/index.html>
    - <https://rest.ensembl.org/>



- Comme l'indique le menu, des API utilisant le langage Perl ont été développées (abandon) ainsi qu'un package R « **BiomaRt** » utilisable avec RStudio et particulièrement intéressant dans le cadre d'applications Shiny.



[Video 1 - biomaRt v1](#)



[Video 2 - Biomart Tutorial](#)

- NCBI : API « Entrez Programming Utilities » (E-utilities)

## APIs

NCBI provides several public APIs that allow programmatic access to many databases and tools.

### Entrez Programming Utilities (E-utilities)

The E-utilities are the public API to the NCBI Entrez system and allow access to all Entrez databases including PubMed, PMC, Gene, Nucleotide and Protein. The E-utilities are a suite of eight server-side programs that accept a fixed URL syntax for search, link and retrieval operations. A companion package named Entrez Direct consists of several executables that allow the E-utilities to be called directly from a UNIX command line.

[Documentation](#)   [Quick Start](#)   [Examples](#)   [Entrez Direct](#)

- <https://www.ncbi.nlm.nih.gov/>
    - <https://www.ncbi.nlm.nih.gov/home/develop/api/>
    - <https://www.ncbi.nlm.nih.gov/books/NBK25501/>
    -  [E-utilities Introduction](#)
    - [https://dataguide.nlm.nih.gov/eutilities/how\\_eutilities\\_works.html](https://dataguide.nlm.nih.gov/eutilities/how_eutilities_works.html)
    - Slideshare : <https://www.slideshare.net/mkim8/eutilities>

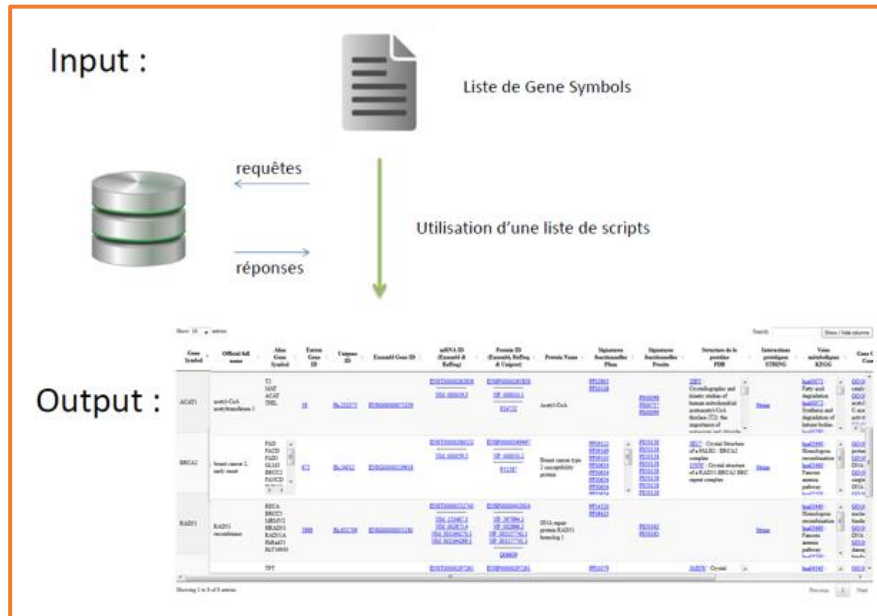
## The 9 E-utilities

- **Introductory Notes**
- **ESearch**: Search a text query in a single Entrez database.
- **ESummary**: Retrieve document summaries for each UID.
- **EFetch**: Retrieve full records for each UID.
- **EPost**: Upload a list of UIDs for later use.
- **ELink**: Retrieve UIDs for related or linked records, or LinkOut URLs.
- **EInfo**: Retrieve information and statistics about a single database.
- **ESpell**: Retrieve spelling suggestions for a text query.
- **ECitMatch**: Search PubMed for a series of citation strings.
- **EGQuery**: Search a text query in all Entrez databases and return the number of results for the query in each database.

- Inconvénient : documentation peu aidante, API initialement basée sur le langage Perl
- **Récemment : API en bioPython :**
  - <https://biopython.org/docs/dev/api/Bio.html#subpackages>
  - <https://biopython.org/docs/dev/api/Bio.Entrez.html>
- **Autres ressources**
  - Uniprot :
    - <https://www.uniprot.org/>
    - <https://www.uniprot.org/help/programmatic%5Faccess>
  - KEGG et reactome
    - <https://www.genome.jp/kegg/>
    - <https://www.kegg.jp/kegg/rest/>
    - <https://reactome.org/>
    - <https://reactome.org/ContentService/>
  - GO :
    - <https://www.ebi.ac.uk/QuickGO/>
    - <https://www.ebi.ac.uk/QuickGO/api/example.html>
    - <http://geneontology.org/>
    - <http://geneontology.org/docs/tools-guide/>
  - Pfam, Prosite :
    - <http://pfam.xfam.org/>
    - <http://pfam.xfam.org/help#tabview=tab11>
    - <https://prosite.expasy.org/>
  - PDB :
    - <https://www.rcsb.org/>
    - <https://data.rcsb.org/#data-api>
  - String :
    - <https://string-db.org/>
    - [https://string-db.org/cgi/access?footer\\_active\\_subpage=apis](https://string-db.org/cgi/access?footer_active_subpage=apis)
  - Etc.

### 3. Cahier des charges du projet « scripting pour l'agrégation automatique d'annotations »

1. Vous mettrez en place une série de scripts permettant d'agréger à partir d'une liste de gènes d'une espèce donnée, leurs annotations respectives dans un fichier tabulé interactif.
2. Optionnel (si 1 fini) : développement d'une interface, en équipe



#### Quelles annotations ?

**En bleu** : données de départ (input)

**En rouge** : sources primaires à utiliser

En noir, liste des informations requises dans le tableau final (output)

#### ☒ Informations générales : ADN-ARN(s)-Protéine(s)

- ✓ **Gene Symbol** (ex : RAD51 sur **Gene**, **NCBI**) pour l'organisme **Genre species** (ex : *Homo sapiens*)
- ✓ Official full name (Ex : RAD51 recombinaison, **Gene**, **NCBI**)
- ✓ Gene access number : (ex : 5888 sur **Gene**, **NCBI** ; sur **Ensembl**: (ex : ENSG00000051180) + le lien de visualisation sur le **genome browser Ensembl et UCSC**)
- ✓ RNA access number(s) (ex : NM\_001164269.1. sur **RefSeq**, ENST00000267868 sur **Ensembl**)
- ✓ Protein name(s) (ex : DNA repair-protein RAD51 homolog 1 sur **UniprotKB**)
- ✓ Protein acces number(s) (ex : Q06609 sur **UniprotKB**, NP\_001157741.1. sur **RefSeq**, ENSP0000026786 sur **Ensembl**)

#### ☒ Annotation fonctionnelle et structurale de la protéine

- ✓ Functional signature(s) (ex : PF08423 domaines protéiques sur **Pfam** + graphical view, ex : PS50162 motifs et domaines sur **Prosite** + graphical view)
- ✓ 3D Protein Structure(s) : (ex : 1b22, N-terminal Domain sur **PDB** ou autre banque de structures)

Annotation relationnelle :

- ✓ Gene ontologies (ex : Function : transcription; Biological process : DNA damage ; Cellular component : nucleus sur **GO** ou **Quick GO**)
- ✓ Pathways (ex : hsa:5888 + les voies hsa03440 Homologous recombination sur **KEGG** ou **Reactome**)
- ✓ Protein Interactions (ex : lien vers **String** ou **IntAct**)
- ✓ Orthologous gene(s) (**Ensembl Compara**)

## Contraintes de la solution

- ✓ Le modèle de la collecte des annotations est fourni (annexe - figure 1). Vous aurez à redessiner le vôtre, le précisez avec le nom de vos méthodes et scripts. L'origine de l'annotation est dès que possible la source primaire de l'information.
- ✓ Pour agréger les annotations, vous utiliserez les connaissances acquises au cours de vos enseignements en réinvestissant obligatoirement une **diversité d'outils de scripting et web services** : programmation Python et bioPython; API (API REST Ensembl, API e-utilities du NCBI), HTML et construction d'URL, etc.. Vous aurez à utiliser également le format JSON et XML.
- ✓ Organisation des scripts (annexe - figure 2).
  - Organisation **modulaire** selon les bases de données
  - Lancement par un script principal (main.py)
  - Structure de données pour la collecte des annotations
- ✓ Votre solution devra fonctionner **quelle que soit l'espèce (fichier input fourni)**
- ✓ Le tableau interactif sera construit avec le **plug-in DataTables de la librairie jQuery en Javascript**:  
<https://www.datatables.net/>  
Il comportera donc les liens HTML fonctionnels vers les banques ressources et les autres facilités d'utilisation interactive (fonction de tri, recherche, ascenseur, personnalisation de l'ordre et de l'affichage des colonnes...). (annexe – figure 3 et exemple de fichiers output)

## 4. Modalité

- ✓ **Phase individuelle** : elle s'effectue en lien avec les enseignements qui suivent. A rendre pour fin janvier (date à préciser). Suivi et évaluation assurée par les enseignants d'informatique (AL, TL)
  - Module Ensembl (réalisé quasiment en TP)
  - Module NCBI

### Livrables sur Moodle :

- Deux fichiers : `ncbi.py` et `ensembl.py`
- Le fichier input comportant un exemple d'un petit jeu de données (10 Gene symbols) de la forme `Genesymbols.txt`

- ✓ **Phase collective** : 3 groupes de n (X BIMS + 1 CCB4). Livraison pour semaine X (date à préciser)
  - Choix des modules Ensembl et NCBI parmi le groupe
  - Développement des autres modules –répartition dans l'équipe
  - Interface

### ➤ Livrable sur Moodle :

Un fichier de la forme `NomGroupe_Annotation.tar.gz` comprenant :

- Le fichier final de votre modélisation de la forme `NomGroupe_schema_conceptuel.pdf`
- Le diaporama de votre présentation (Cf ci-dessous) `NomGroupe_Presentation.pdf`
- Les sources de vos scripts de la forme `script.py`
- Le fichier input comportant un exemple d'un petit jeu de données (10 Gene symbols) de la forme `Genesymbols.txt`
- Un fichier output correspondant de la forme `Results.html`
- (Option) L'application interfacée déployable

- **Présentation orale par équipe** : Semaine X (créneaux à préciser) ; 10 min + 5 min questions (HD et TL)

Pas d'introduction !

1. Votre solution (livrable) :
  - **D1 : Modélisation** > schéma général de votre collecte d'informations d'annotations avec les méthodes
  - **D2 : Organisation du livrable**, ie du fichier source (fichier input, organisation des modules, script principal, fichier sortie) ;
  - D3 : lancement de la démonstration en ligne de code et sur l'interface (si réalisée)
2. **[D4-D8 max] Présentation de portions de votre code**: un exemple pour chaque type de solution de script (API, e-utilities construction d'URL, JSON/XML etc.) avec arguments et paramètres ; structure de données collectées et création du tableau de sortie
3. (Option) D9 Présentation technique de l'interface
4. **D10 (max) Conclusions/ Autoréflexions**: difficultés et contournements, points positifs/négatifs (complet ou pas, temps d'exécution...).